Software libre



Roger Baig Viñas
Francesc Aulí Llinàs

71Z799002MO

Sistema operativo GNU/Linux básico

UOC
Formación de Posgrado

David Megías Jiménez

Coordinador

Ingeniero en Informática por la UAB.

Magíster en Técnicas Avanzadas de Automatización de Procesos por la UAB.

Doctor en Informática por la UAB.

Profesor de los Estudios de Informática y Multimedia de la UOC.

Jordi Mas

Coordinador

Coordinador general de Softcatalà y desarrollador del procesador de textos libre Abiword.

Miembro fundador de Softcatalà y de la red telemática RedBBS.

En calidad de consultor, ha trabajado en empresas como Menta, Telépolis, Vodafone, Lotus, eresMas, Amena y Terra España.

Roger Baig i Viñas

Autor

Ingeniero Técnico Superior Industrial (UPC) e Ingeniero en Electrónica y Automática Industrial (UPC). Profesor asociado Departamento de Telecomunicaciones e ingeniería de sistemas (UAB)

Francesc Aulí Llinàs

Autor

Ing. Informática (UAB) - Premio extraordinario. Concesión beca FPI (Generalitat de Catalunya)

Primera edición: noviembre 2003

Av. Tibidabo, 39-43, 08035 Barcelona
Material realizado por Eureca Media, SL

Depósito legal: B-38.683-2003
ISBN: 84-9788-028-3

Índice

Introducción

Aunque ya hace más de veinte años que el software libre existe, hasta los últimos tiempos no se ha perfilado como una alternativa válida para muchos usuarios, empresas y, cada vez más, instituciones y gobiernos. Actualmente, GNU/Linux es uno de los sistemas operativos más fiables y eficientes que podemos encontrar. Aunque su naturaleza de software libre creó inicialmente ciertas reticencias por parte de usuarios y empresas, GNU/Linux ha demostrado estar a la altura de cualquier otro sistema operativo existente.

El objetivo de este curso es iniciarnos en el mundo del GNU/Linux. En él obtendremos las claves para entender la filosofía del código libre, aprenderemos cómo usarlo y manipularlo a nuestro gusto y dispondremos de las herramientas necesarias para poder movernos fácilmente en este nuevo mundo. El documento tampoco pretende ser un manual de referencia imprescindible para administradores y/o usuarios; para ello ya existen centenares de manuales, HOWTOS y multitud de otras referencias que nos ocuparían millares de páginas. Aquí pretendemos aprender a dar los primeros pasos en esta tierra poco explorada aún para demasiados usuarios y administradores, a la vez que enseñaremos cómo plantear y resolver por nosotros mismos los problemas que puedan aparecer.

El curso no pretende basarse en ninguna distribución en particular, pero en la mayoría de ejemplos y actividades es necesario concretar específicamente algunas acciones y se utilizará Debian GNU/Linux (versión 3.0 -Woody-). Aunque no es una distribución tan intuitiva y fácil de utilizar como algunas otras, nos servirá para explicar paso a paso todas las características de un sistema operativo basado en GNU/Linux. Además, su extraordinaria calidad, estabilidad y seguridad la hacen una de las opciones actualmente más válidas. Por otra parte, tampoco debemos olvidar el soporte (Debian está desarrollada por voluntarios y no da ninguna clase de soporte) que se da en otras distribuciones y que en el caso de muchas empresas es imprescindible. Por esta razón, hemos incluido un apéndice donde mostra-

ANOTACIONES

mos el proceso de instalación y las principales características de RedHat Linux (versión 9.0).

Esperamos que el curso sea de su agrado y sirva para abrirle las puertas al mundo del software libre. Cuantos más usuarios seamos, más software y de mejor calidad tendremos.

¡Bienvenidos al GNU/Linux!

1. Presentación

1.1. ¿Qué es el GNU?

Para entender todo el movimiento del software libre, debemos situarnos a finales de la década de los sesenta, principios de los setenta. En aquellos tiempos las grandes compañías de ordenadores no daban el valor que hoy día se da al software. En su gran mayoría eran fabricantes de ordenadores que obtenían sus principales ingresos vendiendo sus grandes máquinas, a las que incorporaban algún tipo de sistema operativo y aplicaciones. Las universidades tenían permiso para coger y estudiar el código fuente del sistema operativo para fines docentes. Los mismos usuarios podían pedir el código fuente de *drivers* y programas para adaptarlos a sus necesidades. Se consideraba que el software no tenía valor por sí mismo si no estaba acompañado por el hardware que lo soportaba. En este entorno, los laboratorios Bell (AT&T) diseñaron un sistema operativo llamado **UNIX**, caracterizado por la buena gestión de los recursos del sistema, su estabilidad y su compatibilidad con el hardware de diferentes fabricantes (para homogeneizar todos sus sistemas). Este último hecho fue importantísimo (hasta entonces todos los fabricantes tenían sus propios operativos incompatibles con los otros), ya que devino el factor que le proporcionó mucha popularidad.

Poco a poco, las grandes empresas empezaron a tomar conciencia del valor del software: primero fue IBM la que en 1965 dejó de dar el código fuente de su sistema operativo, a finales de los setenta Digital Research empezó a vender el suyo, etc. Este hecho hizo que todas las compañías se dieran cuenta de que el software podía ser muy rentable y les podía aportar grandes beneficios. A partir de este hecho, la mayoría de empresas empezaron a poner reticencias a dejar el código fuente de sus programas y sistemas operativos y empezaron a vender sus programas como un valor añadido a su hardware. En este entorno cada vez más cerrado, **Richard Stallman** (que trabajaba en el MIT, Massachusetts Institute of Technology) se sintió indignado al comprobar que cada vez era más difícil conseguir el código

Nota

El mismo Stallman cuenta como anécdota lo mucho que se enfadó al descubrir que la compañía que les había vendido una nueva impresora para el laboratorio donde trabajaba no le quería facilitar el código fuente de los drivers. ¡Él sólo quería modificarlos para que le avisara automáticamente cuando se atascaba el papel! La compañía se negó a proporcionárselos.

fuente de los programas que utilizaba para adaptarlos a sus necesidades, tal como había hecho hasta entonces.

A partir de ese momento, Stallman decidió ser consecuente con sus ideales e iniciar un gran proyecto para intentar abrir otra vez el código fuente de los programas. Consciente de que no podría conseguir que las compañías cedieran en este punto, se propuso crear su propio sistema operativo y aplicaciones iniciando un proyecto llamado GNU.

Contenido complementario

El nombre que le dio al proyecto significa GNU, Not UNIX, añadiéndose a la moda de los nombres/bromas recursivas de aquel tiempo.

De especial interés para entender los motivos que llevaron a Stallman a iniciar GNU es su primer manifiesto, el documento donde explicó a toda la comunidad en qué consistiría el proyecto, cómo lo orientaría y por qué tenía que hacerlo. En él empezó a describir el concepto de software libre y para qué creía necesario que programadores y desarrolladores de alrededor del mundo contribuyeran con él. Aunque en muchas ocasiones se confunde el concepto de software libre con el de software gratuito (en inglés, free tiene los dos significados), en posteriores documentos se ha dejado muy claro que el software libre no debe por qué ser gratuito. Debemos entender como software libre programas de los cuales podemos conseguir su código fuente, estudiarlo, modificarlo y redistribuirlo sin que nos **obliguen** a pagar por ello. Lo que debemos tener claro es que sí que podemos pedir el dinero que queramos por los programas y su código fuente, el soporte que podemos ofrecer a los usuarios, los libros que vendamos o el material que proporcionemos, tal y como muchas compañías que distribuyen GNU/Linux hacen. Sin embargo, en ningún momento, podemos obligar a que los usuarios no distribuyan el software que les hemos vendido. Éste debe poder ser distribuido de forma libre. Es una forma diferente de entender el software a la que estamos acostumbrados. En muchos de los textos de la FSF (Free Software Foundation) se habla más de filosofía que de ingeniería. Debemos entender todo este movimiento más como una forma de pensar o hacer las cosas que como una compañía más de software.

La filosofía que en la FSF se tiene del software lo define con las siguientes cuatro libertades:

- La libertad 0 se refiere a la libertad de poder usar el programa para cualquier propósito.

ANOTACIONES

- La libertad 1 es la que permite estudiar cómo funciona el programa y adaptarlo a las propias necesidades. El acceso al código fuente es una condición necesaria para garantizar esta libertad.

- La segunda libertad es la que permite distribuir libremente copias del software, ayudando al vecino.

- La última libertad es la que permite mejorar el programa y hacer públicas las propias mejoras, en beneficio de toda la comunidad. El acceso al código fuente, asimismo, es un requisito imprescindible para asegurar esta libertad.

Para dar todas estas libertades al software que se desarrollaba en el proyecto y a los usuarios finales del mismo se escribió la licencia, con la cual se ha protegido todo este tipo de programas, la GPL (General Public License). Esta licencia pone por escrito las ideas anteriormente comentadas.

El proyecto empezó a producir software a partir de 1984, comenzando con el desarrollo de todas la herramientas necesarias para poder implementar un sistema operativo completo. Aunque realizar un proyecto de estas características es un proceso largo y complejo, desde el principio muchos programadores y desarrolladores de software se vieron cautivados por la idea de Stallman y empezaron a colaborar con él de forma gratuita. La comunidad no paró de crecer, y poco a poco empezaron a disponer de las herramientas necesarias (editores, compiladores, etc.) para implementar el núcleo del sistema operativo, que era la tarea que requería las herramientas que se estaban desarrollando. Desde el primer momento se quiso crear un sistema operativo parecido a UNIX y siguiendo las normas POSIX (*Portable Operating System Interface*). Si bien UNIX también tenía sus problemas y carencias, era, y sigue siendo, suficientemente bueno como para adaptarse a la mayoría de las necesidades. La tarea de diseñar y escribir el núcleo del sistema operativo fue la que se dejó para el final del proceso. Aún actualmente está por finalizar definitivamente y el núcleo del GNU, llamado Hurd, permanece en fase de desarrollo.

Contenido complementario

Como su nombre indica, el núcleo (kernel) de un sistema operativo es el corazón con el cual puede funcionar. Es el núcleo de software que gestiona los recursos del ordenador: se comunica con los dispositivos y aplicaciones instalados, administra la memoria adecuadamente, reparte tiempo de procesamiento para todos los programas, se comunica con los dispositivos de almacenamiento para guardar los archivos, etc.

ANOTACIONES

Actividades

1. Leer el primer mensaje escrito por Stallman en 1983 anunciando su proyecto (traducido al castellano): http://www.fsf.org/gnu/initial-announcement.es.html

2. Leer "El Manifiesto GNU" original de Stallman (traducido al castellano): http://www.fsf.org/gnu/manifesto.es.html

3. Leer la "General Public License": http://www.fsf.org/licenses/gpl.html

1.2. ¿Qué es el GNU/Linux?

En este contexto, y cuando la FSF todavía no tenía ningún núcleo estable para su sistema operativo, un profesor de la Universidad de Holanda, **Andrew Tanenbaum**, decidió escribir un sistema operativo para que sus estudiantes pudieran estudiarlo. Igual que Stallman, hasta el momento había podido utilizar el código fuente del UNIX de AT&T para que sus alumnos aprendieran a diseñar sistemas operativos. Su idea era escribir un sistema operativo que pudiera ser estudiado y modificado por cualquiera que quisiera. En 1987 se puso manos a la obra y llamó a su proyecto mini UNIX, dando lugar a **MINIX**. Al no utilizar ni una sola línea de código del UNIX de AT&T, no hay ninguna restricción en coger el código, utilizarlo y modificarlo libremente.

Tanenbaum quiso crear un sistema orientado a fines docentes, por lo que lo diseñó utilizando una arquitectura *micro-kernel*, ideal para una fácil comprensión y aportando una tecnología muy novedosa para la época que le permitía versatilidad, multiplataforma, etc. Éste ha sido uno de los puntos fuertes y débiles a la vez del MINIX: aunque el sistema es una pequeña joya para su estudio y diseño, es muy probable que nunca se pueda utilizar en entornos reales. Se optó por hacerlo entendedor, modular y muy pedagógico, pero no rápido. De todas formas, Tanenbaum tampoco pretendía eso; a lo largo de los años MINIX ha ido evo-

Contenido complementario

La tecnología micro-kernel se basa en dividir las diferentes funcionalidades del núcleo de un sistema operativo en programas totalmente separados y que se comunican entre sí. Esto lo hace muy modular, facilitando muchísimo el test, detección y corrección de errores, mantenimiento, etc. Actualmente, algunos sistemas operativos como Amoeba, Chorus, Mach o WindowsNT™ han incorporado este tipo de tecnología.

ANOTACIONES

lucionando y realmente hoy en día todavía sigue existiendo y siendo estudiado por muchos alumnos de universidades de todo el mundo.

Aquí es cuando entra en juego **Linux**. Mientras la FSF seguía con su gran proyecto proporcionando herramientas para la construcción de un sistema operativo, Tanenbaum orientaba MINIX para fines docentes y muchas empresas seguían haciendo evolucionar sus propias versiones de UNIX. **Linus Torvalds**, estudiante de la Universidad de Helsinki, decide crear en agosto de 1991 su propio núcleo para un nuevo sistema operativo, Linux. Su idea era crear un UNIX para PC para que todos los que quisieran lo pudieran utilizar en su ordenador. La primera aparición en escena que hizo fue en un debate sobre MINIX y sistemas operativos, donde expuso las siguientes ideas:

```
Newsgroups: comp.os.minix
Asunto: What would you like to see most in minix?
Fecha: 25 Aug. 91 20:57:08 GMT
Organization: University of Helsinki
Hello everybody out there using minix.
I'm doing a (free) operating system (just a ho-
bby, won't be big and professional like gnu) for
386(486) AT clones. This has been brewing since
april, and is starting to get ready. I'd like any
feedback on things people like/dislike in minix,
as my OS resembles it somewhat (same physical la-
yout of the file-system (due to practical rea-
sons) among other things).
I've currently ported bash(1.08) and gcc(1.40),
and things seem to work.
This implies that I'll get something practical
within a few months, and I'd like to know what
features most people would want. Any suggestions
are welcome, but I won't promise I'll implement
them :-)
```

Si accediéramos al fórum de debate donde apareció este primer mensaje, veríamos cómo rápidamente gente de todo el mundo empezó a interesarse por este nuevo sistema, que al utilizar el compi-

Contenido complementario

Linux, el núcleo de GNU/Linux, es de tipo monolítico. Esto indica que no se separan sus diferentes funcionalidades en distintos módulos, sino que todo forma parte de un mismo programa. El principal inconveniente de este tipo de diseño es que la localización de errores y su mantenimiento son muy costosos. En contrapartida, el rendimiento que se consigue es mucho mayor que en otros tipos de diseño.

lador e intérprete de comandos de GNU (gcc y bash) como piezas fundamentales, también tenía las características de software libre. Aunque en palabras del mismo Torvalds, si él hubiera sabido la cantidad de trabajo necesario para lograr que su idea funcionase, nunca lo hubiera hecho: esfuerzos de muchos expertos en informática de todo el mundo hicieron posible este proyecto.

De hecho, en los primeros años de su existencia, GNU/Linux se identificaba como el sistema operativo de los *hackers*. Su difícil instalación, manipulación y falta de *drivers* lo hacían una herramienta apta únicamente para gente muy entendida en el tema. Fueron estos primeros usuarios los que diseñaron los *drivers* para los discos, impresoras, tarjetas, etc. y los que empezaron a dar a conocer al mundo este sistema. Poco a poco, el número de usuarios empezó a crecer y actualmente ya existen muchas empresas y grupos de usuarios que crean sus propias distribuciones de GNU/Linux.

1.3. Distribuciones

Actualmente, existen muchas distribuciones diferentes basadas en GNU/Linux. Las hay para toda clase de ordenadores y dispositivos electrónicos: ordenadores portátiles o de sobremesa, pocketPC o PDA, puntos de acceso de redes *wireless*, etc. La naturaleza del software libre permite esto: cualquiera puede coger el código desarrollado hasta el momento y adaptarlo a sus propias necesidades. Es un hecho que, cada vez más, empresas y usuarios eligen sistemas basados en GNU/Linux por sus elevadas prestaciones y la cantidad de software disponible.

De todos modos, aunque existen decenas de distribuciones, hay algunas más populares que se han extendido mucho. La filosofía de software libre hace que muchas empresas que han creado sus propias distribuciones de GNU/Linux no restrinjan el acceso a su código. Aun así, el soporte que ofrecen y el material que venden les aporta beneficios, permitiendo su subsistencia. Asimismo cabe considerar que en muchas de estas distribuciones se incluye software propietario que algunos usuarios prefieren, si bien en muchos casos existen programas homólogos con licencia Free Software.

Contenido complementario

Aunque muchas distribuciones de GNU/Linux se denominan solamente Linux, es importante que diferenciemos que realmente Linux es el núcleo del sistema operativo y que el proyecto GNU es el que realmente ha aportado mucha de la estructura para el funcionamiento del mismo.

A continuación haremos una breve descripción de algunas de las distribuciones de GNU/Linux:

- Slackware: una de las primeras distribuciones que aparecieron. Fue creada por Patrick Volkerding y tuvo un gran éxito en sus primeros años de existencia.

- Debian GNU/Linux: una de las primeras distribuciones de GNU/Linux que aparecieron y aún siguen existiendo y evolucionado. El sistema de paquetes nos permite diferenciar claramente el software libre del que no lo es, permitiéndonos disponer de todo el sistema solamente con programas de licencia Free Software. Está desarrollada por un grupo de colaboradores distribuidos por todo el mundo y no cuenta con el respaldo de ninguna empresa. Aunque es de las más estables y seguras que existen, su sistema de instalación y configuración necesita de conocimientos previos.

- RedHat Linux: junto con SuSE, es una de las distribuciones de mayor popularidad. Está creada por una empresa de EUA, aportando software de gran calidad. Tiene un entorno muy intuitivo que facilita mucho su instalación y configuración.

- SuSE Linux: aunque es una distribución creada bastante recientemente, ha tenido una gran difusión. Está desarrollada por una

ANOTACIONES

empresa alemana, aportando mucho software propietario de calidad. Es muy completa y fácil de instalar y mantener, aunque en algunos aspectos no se siguen algunos de los estándares de la comunidad.

- Knoppix: distribución en un CD-live basada en Debian. Detecta automáticamente todo tipo de hardware y aporta el último escritorio de KDE y la suite OpenOffice.org. Muy útil para demostraciones y usuarios noveles en el sistema.

Tampoco podemos olvidar que existen otros sistemas operativos compatibles con UNIX y los estándares que se siguen actualmente. Muchos de los conceptos y herramientas que veremos a lo largo del curso también servirán para estos otros. En especial debemos destacar GNU/Hurd (núcleo desarrollado por el proyecto GNU) y FreeBSD.

Actividad

4. Leer la descripción de algunas de las distribuciones actuales basadas en GNU/Linux: http://www.linuxhq.com/dist.html

1.4. Programas y documentación

Internet ha sido siempre el principal medio de comunicación entre los desarrolladores y usuarios del software libre. Es por esta razón por lo que ya desde el principio de la gran expansión de GNU/Linux se ha podido encontrar en la Red muchísima información sobre el operativo. La mayoría de los programas los podemos descargar de Internet, em-

paquetados con alguno de los sistemas más comunes o bien directamente a partir de su código fuente para que lo podamos compilar en nuestro sistema. Además, la mayoría de las distribuciones también se pueden descargar de la Red sin necesidad de comprar ningún *pack* especial de las revistas especializadas o de las mismas empresas que lo producen. También es cierto que si queremos el soporte que ofrecen algunas de las distribuciones, lo mejor es comprar todo el material que se proporciona (CD, manuales, etc.) y registrarse.

A medida que nos vayamos introduciendo en el mundo del software libre y del GNU/Linux, veremos cómo uno de los aspectos clave para moverse por él es saber encontrar la documentación que nos interesa. Cuando nos encontramos ante un problema, antes de empezar a dar vueltas sobre cómo resolverlo, debemos pensar que es muy probable que otra gente como nosotros se haya encontrado con lo mismo o con algo similar. Buscar y encontrar la documentación que se adapte mejor a los problemas que se nos vayan planteando nos ahorrará mucho tiempo y esfuerzo. La comunidad del software libre genera centenares de documentos que podemos descargarnos libremente de Internet, además de los foros de discusión, páginas de rumores y noticias, etc.

Algunas de las referencias más populares y que más nos pueden ayudar son:

- **Documentación**

http://www.tldp.org: The Linux Documentation Project. La mayoría de guías, HOWTOS, FAQS, etc. existentes las podemos encontrar en este sitio, que además está en varios idiomas.

http://lucas.linux.org.mx: LinUx en CAStellano. Gran proyecto de documentación en castellano para los HOWTOS, guías, etc. de GNU/Linux.

http://www.linuxpowered.com/HOWTO/HOWTO-INDEX: El HOWTO de los HOWTOS.

http://www.linux.com: Página con diferentes secciones de noticias, documentación, etc.

http://www.debian.org/doc: Documentación para Debian GNU/Linux.

ANOTACIONES

- Noticias

http://slashdot.com: Noticias y rumores del mundo GNU/Linux. En inglés.

http://barrapunto.com: La réplica de slashdot en castellano.

http://puntbarra.com: La réplica de slashdot en catalán.

http://bulmalug.net: Bisoños usuarios de Linux de Mallorca y alrededores. Noticias

y secciones dedicadas a temas concretos.

http://www.es.gnu.org/gnuticias: Noticias de GNU en español.

http://linuxtoday.com: Otra página de noticias muy práctica para estar a la última.

http://libertonia.escomposlinux.org: Página de noticias. De especial interés es su sección de "Fuentes de Noticias", donde hay multitud de otros enlaces a otras páginas del mismo estilo.

- Foros

http://www.foroslinux.org: Varios foros de GNU/Linux dedicados a todo tipo de temas.

http://www.linuxsecurity.com/resources/forums-1.html: Foros centrados en temas de seguridad y similares.

- Búsqueda

http://www.google.com/linux: El mayor buscador del mundo también para GNU/Linux.

http://www.buscadoc.org: Buscador de documentación informática en castellano.

ANOTACIONES

- Distribuciones

http://www.fsf.org: La página oficial de la Free Software Foundation.

http://www.debian.org: Página oficial de debian GNU/Linux.

http://www.redhat.com: Página oficial de RedHat Linux.

http://www.suse.com: Página oficial de SuSE.

http://www.slackware.com: Página oficial de Slackware Linux.

http://www.knoppix.com: Página oficial de Knoppix.

- Descargas

http://sourceforge.net: La mayor página con proyectos de software libre.

http://www.softonic.com/index.phtml?n id=4: Sección de descarga para GNU/Linux de una de las múltiples páginas de *downloading*.

http://download.com: Página de descargas.

- Otras

http://www.linuxsecurity.com: Página muy actual centrada en todo tipo de temas de seguridad en GNU/Linux.

http://www.linuxhq.com: Información general sobre distribuciones de GNU/Linux, seguridad, etc.

http://www.linuxjournal.org: Página de noticias y artículos sobre GNU/Linux.

http://www.linuxgazette.com: Revista de GNU/Linux.

http://www.linux-mag.com: Revista de GNU/Linux.

http://www.xfree86.org: Página oficial del proyecto XFree86.

ANOTACIONES

2. Conceptos y comandos básicos

2.1. Introducción

En este capítulo aprenderemos las ideas e instrucciones básicas para movernos adecuadamente por el sistema. Si no estamos acostumbrados a utilizar la línea de comandos para manipular el sistema operativo, al principio puede parecernos un poco complicado, pero a medida que las vayamos utilizando veremos que son muy útiles y nos permiten realizar cualquier tarea que queramos hacer. Además, el hecho de saber utilizar correctamente los comandos nos será muy útil cuando necesitemos conectarnos de forma remota a una máquina y podremos diseñar, asimismo, pequeños programas (*shell scripts*) para automatizar las tareas de administración más comunes.

Contenido complementario

Un comando es un programa que realiza una determinada acción relacionada con el sistema operativo.

La mayoría de los comandos que veremos en este capítulo forman parte del estándar (normas IEEE POSIX) y son comunes a todos los sistemas GNU/Linux y a UNIX. Aunque cada distribución tiene sus propias aplicaciones de administración y gestión, muchas de las acciones que se realizan a partir de ellas también se pueden hacer con los comandos que veremos. A partir de los mismos, podremos manipular casi todos los aspectos del sistema y movernos eficientemente por él. Aprendiendo a utilizar correctamente estos comandos, aprenderemos a navegar por cualquier sistema basado en GNU/Linux, sin importar qué distribución estemos usando.

Cada uno de los comandos del sistema suele tener multitud de parámetros diferentes. Con la utilización de los parámetros podemos, con un mismo comando, hacer muchas acciones diferentes, aunque todas sean de un mismo estilo. En este documento no especificaremos los diferentes parámetros de cada uno de los comandos que veremos, ya que extenderíamos el texto más allá de lo permisible y tampoco tiene sentido conocer exactamente la totalidad de los parámetros posibles para cada uno. Todos ellos disponen de un amplio manual, donde se especifican todas sus opciones, de manera que siempre que necesitemos realizar alguna acción en concreto podre-

Contenido complementario

Un parámetro no es más que una opción determinada de un comando, que añadimos a continuación del mismo, precedido por un espacio y, en muchas ocasiones, por un guión. Por ejemplo, si un comando fuera listar, podríamos pasarle un parámetro como "`listar -todo`".

mos recurrir a él. En los talleres distribuidos a lo largo del curso sí que veremos algunas de estas opciones, aunque es importante saber que con el manual siempre podremos descubrir muchas otras, que nos pueden ayudar a realizar todo lo que necesitemos.

2.2. Usuarios y grupos

Nota

Una política de nombres muy utilizada suele ser poner como login la primera inicial del nombre del usuario seguido de su apellido.

Actualmente, la mayoría de los sistemas operativos existentes son multiusuario y multitarea. Ello implica que más de un usuario puede trabajar en el sistema de forma simultánea a otros, ejecutando una o más tareas a la vez. Por este motivo, es muy importante que el mismo sistema operativo incorpore mecanismos para manipular y controlar correctamente a los usuarios: el sistema de entrada e identificación (*login*), los programas que puede ejecutar cada uno, mecanismos de seguridad para proteger el hardware del ordenador, protección para los ficheros de los usuarios, etc.

Contenido complementario

NIS son una serie de aplicaciones que nos permiten gestionar todos los usuarios de un mismo laboratorio de forma centralizada en un solo servidor.

Los sistemas operativos basados en UNIX organizan toda esta información por usuarios y grupos. Al entrar en el sistema, debemos identificarnos con un *login* y una contraseña. El *login* suele ser un nombre que identifica de forma inequívoca al **usuario**. En sistemas donde hay más que unos pocos usuarios, es importante disponer de una buena política de nombres para poderlos identificar a todos de forma clara. La contraseña debe ser una combinación de letras, números y caracteres especiales. No debe estar formada por ninguna palabra de diccionario o similares porque puede representar un problema de seguridad importante. El sistema de contraseñas es de tipo unidireccional. Esto quiere decir que nuestra contraseña no es almacenada como texto, sino que es cifrada y guardada tal como es. Cuando entramos en el sistema y escribimos nuestra contraseña, ésta es cifrada y comparada con la que está almacenada. Si coinciden, la identificación es positiva, si no coinciden, no hay identificación. Lo importante de todo este sistema es que a partir del cifrado no podemos conseguir, de ninguna manera, la clave original. Los programas que intentan romper las contraseñas de los usuarios lo único que pueden hacer es cifrar palabras a partir de diccionarios (con sistemas automáticos para derivarlas y buscar variantes) y probar si coinciden con el cifrado de alguna de las contraseñas de usuario. Es por este motivo por lo que debemos escoger cuidadosamente

ANOTACIONES

nuestras contraseñas; de otra forma comprometeremos toda la seguridad del sistema.

Actualmente, en los sistemas GNU/Linux podemos escoger dos tipos de cifrado posibles para las contraseñas de usuario. El que se viene usando desde los inicios de UNIX es el 3DES. El único inconveniente de este tipo de cifrado es que sólo nos permite contraseñas de 8 letras (si escribimos más, se ignoran), a diferencia del otro tipo de cifrado, llamado MD5, con el que podemos usar contraseñas de la longitud que queramos (de hecho, MD5 es un sistema de *hashing*, pero también se puede utilizar para cifrar contraseñas de forma unidireccional). Cuanto más larga sea la contraseña, más segura resulta, con lo cual, se recomienda utilizar el segundo tipo de cifrado. De todos modos debemos considerar que, si necesitamos usar algunos programas especiales para la gestión de usuarios, como el NIS, puede que no sean compatibles con MD5.

Si bien un usuario es un individuo particular que puede entrar en el sistema, un **grupo** es un conjunto de usuarios con acceso al sistema que comparten unas mismas características, de forma que nos es útil agruparlos para poder darles una serie de permisos especiales en el sistema. Un usuario debe pertenecer, al menos, a un grupo, aunque puede ser de más de uno. El sistema también utiliza todo este mecanismo de usuarios y grupos para gestionar los servidores de aplicaciones instalados y otros mecanismos. Por esta razón, además de los usuarios reales, en un sistema habrá muchos otros vinculados a otras tareas que se deben realizar en el operativo. Generalmente, este tipo de usuario no podrá entrar (con un *login* normal) al sistema.

Contenido complementario

Un servidor es un programa que se encarga de proporcionar algún tipo de servicio (como servir páginas web, dejar que los usuarios se conecten remotamente, etc.), generalmente vinculado a la Red.

En todo sistema operativo debe haber un superusuario (*root*). Éste será el usuario que contará con todos los permisos, el que tendrá los privilegios máximos que le permitirán efectuar cualquier operación sobre el sistema. Es necesario que éste exista, ya que será quien se encargará de toda la administración y gestión de servidores, grupos, etc. Esta cuenta no debe utilizarse para trabajar normalmente en el sistema. Sólo deberíamos entrar como *root* cuando sea realmente necesario, utilizando otras cuentas para el trabajo normal de los usuarios. De este modo nunca podremos dañar el sistema con operaciones erróneas o con la prueba de programas maliciosos, etc.

ANOTACIONES

Toda la información de usuarios y grupos se guarda en los siguientes archivos:

- `/etc/passwd`: información (nombre, directorio *home*, . . .) del usuario.

- `/etc/group`: información sobre los grupos de usuarios.

- `/etc/shadow`: contraseñas cifradas de los usuarios y configuración para su validez, cambio, etc.

Utilizar el archivo de `shadow` es opcional. En un principio, las contraseñas cifradas de los usuarios se guardaban en el mismo fichero de `passwd`, pero, por razones de seguridad (muchos mecanismos deben poder leer este fichero, con lo cual era muy fácil hacerse con él e intentar "crackear" las contraseñas) se optó por cambiar este mecanismo para hacer que el fichero de `shadow` sólo fuera accesible para algunos usuarios con privilegios especiales en el sistema. Esta opción es configurable en el proceso de instalación del sistema y suele ser recomendable utilizarla. Todos estos ficheros están organizados por líneas, donde cada una de ellas identifica a un usuario o grupo (dependiendo del fichero). En cada línea hay diversos campos separados por el carácter "`:`". En tareas de administración, es importante saber qué son estos campos, por lo que vamos a explorarlos con un poco más de detalle:

- passwd

1) *Login*: el nombre del usuario. No puede haber dos nombres iguales, aunque sí alguno que coincida con un grupo del sistema.

2) Contraseña cifrada: si no se utiliza el fichero de `shadow`, las contraseñas cifradas se almacenan en este campo. Si utilizamos el fichero de `shadow`, todos los usuarios existentes en este fichero deben existir también en el de `shadow` y en este campo se pone el carácter "`x`".

3) *User* ID: número de identificación del usuario. Es el número con el cual el sistema identifica al usuario. El 0 es el único que está reservado para el *root*.

Contenido complementario

También es posible configurar el sistema para que se utilice un fichero `shadow` para los grupos (en caso de que sea necesario ponerles contraseña). Este fichero se nombraría `/etc/gshadow`. Generalmente, la configuración de contraseñas se indica al instalar el sistema, aunque todo se puede cambiar y adaptar a nuestro gusto utilizando los módulos PAM (*Pluggable Authentication Modules for Linux*), que son los programas que se encargan de todo el sistema de autentificación de usuarios.

Contenido complementario

"Crackear" una contraseña significa conseguir la palabra clave utilizando programas especiales para ello. Estos programas también los usan los administradores de sistemas para descubrir qué usuarios utilizan contraseñas demasiado fáciles de descubrir (las contraseñas buenas no se pueden romper de ningún modo sin utilizar grandes supercomputadoras).

ANOTACIONES

4) *Group* ID: el número de grupo al cual pertenece el usuario. Como un usuario puede pertenecer a más de un grupo, este grupo se denomina *primario*.

5) Comentarios: campo reservado para introducir los comentarios que queramos sobre el usuario. Se suele utilizar para poner el nombre completo o algún tipo de identificación personal.

6) Directorio *home*: el directorio *home* del usuario es donde éste podrá guardar todos sus ficheros. Suelen ponerse todos en alguna carpeta del sistema (generalmente `/home/`) y organizados por grupos.

7) Intérprete de comandos: un **intérprete de comandos** (*shell*) es un programa que se encarga de leer todo lo que escribimos en el teclado y ejecutar los programas o comandos que le indiquemos. Hay decenas de ellos, aunque el más utilizado es, sin duda, el bash (*GNU Bourne-Again SHell*). Si en este campo escribimos `/bin/false` no permitiremos que el usuario ejecute ningún comando en el sistema, aunque esté dado de alta en el mismo.

- **group**

1) Nombre del grupo.

2) Contraseña cifrada: la contraseña de un grupo se utiliza para permitir que los usuarios de un determinado grupo se puedan cambiar a otro o para ejecutar algunos programas con permisos de otro grupo (siempre que se disponga de la contraseña).

3) *Group* ID: número de identificación del grupo. Es el número con el cual el sistema identifica internamente a los grupos. El 0 es el único que está reservado para el grupo del *root* (los administradores).

4) Lista de usuarios: los nombres de los usuarios que pertenecen al grupo, separados por comas. Aunque todos los usuarios deben pertenecer a un determinado grupo (especificado en el cuarto campo del fichero de `passwd`), este campo se puede utilizar para que usuarios de otros grupos también dispongan de los mismos permisos que tiene el que se está referenciando.

ANOTACIONES

- shadow

1) *Login*: debe ser el mismo nombre que se utiliza en el fichero de `passwd`.
2) Contraseña cifrada.
3) Días que han pasado, desde el 1 de enero de 1970, hasta que la contraseña ha sido cambiada por última vez.
4) Días que deben pasar hasta que la contraseña pueda ser cambiada.
5) Días que han de pasar hasta que la contraseña deba ser cambiada.
6) Días antes de caducar la contraseña en el que se avisará al usuario de que debe cambiarla.
7) Días que pueden pasar después de que la contraseña caduque, antes de deshabilitar la cuenta del usuario (si no se cambia la contraseña).
8) Días, desde el 1 de enero de 1970, desde que la cuenta está deshabilitada.
9) Campo reservado.

Contenido complementario

En sistemas UNIX es muy común representar las fechas a partir del número de segundos transcurridos desde el 1 de enero de 1970.

Cuando un usuario entra en el sistema, se le sitúa en su directorio *home* y se ejecuta el intérprete de comandos (*shell*) configurado. De este modo ya puede empezar a trabajar. Sólo el *root* del sistema (o los usuarios de su grupo) tienen permiso para manipular la información de los usuarios y grupos, darlos de alta, de baja, etc. Existen muchos comandos para manipular todo esto. Cada uno de ellos tiene, además, varios parámetros diferentes para gestionar todos los campos que hemos visto anteriormente de forma amena. A continuación mostramos algunos de estos comandos:

Contenido complementario

En sistemas donde hay centenares de usuarios, es usual poner algún tipo de mecanismo para restringir el espacio de disco que puede utilizar cada uno. En los sistemas GNU/Linux este sistema se llama cuota.

- `adduser`: nos sirve para añadir un nuevo usuario al sistema. La forma como éste se añade (si no le especificamos nada) se puede configurar en el fichero `/etc/adduser.conf`. Se le pueden pasar multitud de opciones diferentes para especificar el directorio *home*, el *shell* que hay que utilizar, etc.

- `useradd`: crea un nuevo usuario o cambia la configuración por defecto de los mismos. Este comando y el anterior nos pueden servir para realizar las mismas acciones.

- `usermod`: con este comando podemos modificar la mayoría de los campos que se encuentran en el fichero de `passwd` y `shadow`, como el directorio *home*, el *shell*, la expiración de la contraseña, etc.

- `chfn`: cambia la información personal del usuario, contenida en el campo de comentarios del fichero de `passwd`.

- `chsh`: cambia el *shell* del usuario.

- `deluser`: elimina un usuario del sistema, borrando o guardando todos sus ficheros según los parámetros que le pasemos, haciendo copia de seguridad de los mismos o no, etc. La configuración que se utilizará por defecto con este comando está especificada en el fichero `/etc/deluser.conf`.

- `userdel`: comando con las mismas posibilidades que el anterior.

- `passwd`: nos sirve para cambiar la contraseña de un usuario, la información de expiración de las mismas o para bloquear o desbloquear una determinada cuenta.

- `addgroup`: permite añadir un grupo al sistema.

- `groupadd`: lo mismo que el comando anterior, pero con diferentes parámetros.

- `groupmod`: nos permite modificar la información (nombre y GID) de un determinado grupo.

- `delgroup`: elimina un determinado grupo. Si algún usuario todavía lo tiene como primario, no se podrá eliminar.

- `groupdel`: igual que en el caso anterior.

- `gpasswd`: nos sirve para cambiar la contraseña del grupo.

ANOTACIONES

Contenido complementario

Como vemos, en GNU/Linux tenemos más de una manera para hacer una determinada acción. Ésta es la tónica general que se sigue en el sistema: podemos editar directamente los ficheros y modificarlos nosotros mismos, utilizar algunos de los comandos que existen, creárnoslos nosotros mismos, etc. En definitiva, tenemos la posibilidad de elegir qué es lo que más nos gusta.

Para saber qué usuario somos, podemos utilizar el comando `whoami`, que nos mostrará nuestro *login*. `groups` nos sirve para saber a qué grupos pertenecemos e `id` nos mostrará usuario y grupos. También es interesante poder convertirnos en otro usuario sin tener que salir de la sesión (comando `login` o `su`) o cambiarnos de grupo con el comando `newgrp`. Este último comando debemos utilizarlo sólo cuando no pertenecemos al grupo en cuestión y sabemos su contraseña (que debe estar activada en el fichero de `group`). Si sólo necesitamos los permisos del grupo en cuestión para ejecutar un determinado comando, también podemos utilizar `sg`.

Tal como decíamos anteriormente, GNU/Linux es un sistema operativo multiusuario, por lo que en un mismo momento puede haber varios usuarios conectados al sistema de forma simultánea. Para saber qué usuarios hay en un determinado momento, podemos utilizar el comando `who`, que nos muestra la lista de usuarios dentro del sistema. `w`, además, nos muestra qué es lo que están haciendo. Nos podemos comunicar con ellos utilizando el comando `write`, con el cual aparece el mensaje que hayamos escrito en la pantalla del usuario indicada o `wall`, que escribe el contenido del fichero que hayamos especificado a todos los usuarios dentro del sistema. Para activar o desactivar la opción de recibir mensajes tenemos el comando `mesg`. También podemos hacer un *chat* personal con algún usuario a partir del comando `talk`.

2.3. El sistema de ficheros

2.3.1. La jerarquía del sistema de ficheros

Contenido complementario

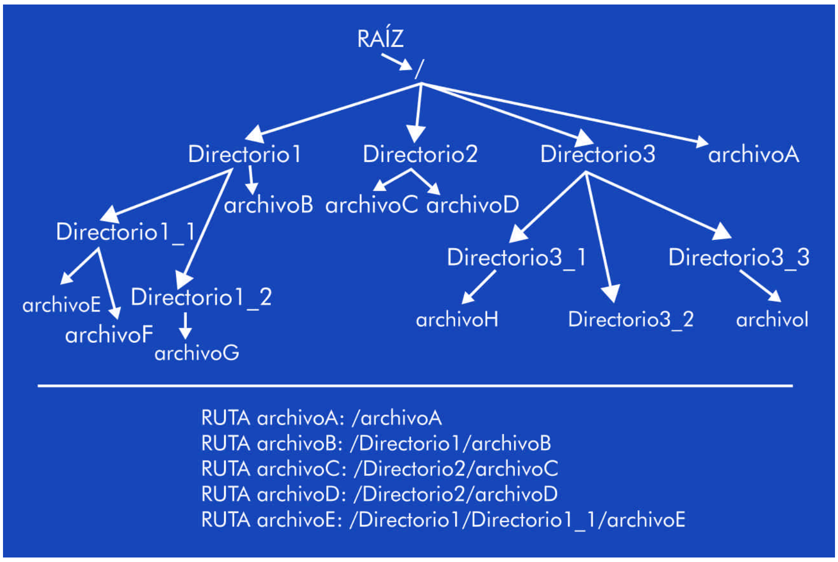
El sistema de ficheros es el programa (o módulos del núcleo del operativo) que se encarga de realizar todas las operaciones relacionadas con el almacenamiento y manipulación de los archivos. Son las funciones que tratan con los dispositivos físicos de almacenamiento del ordenador, como el disco duro.

Todo sistema operativo necesita guardar multitud de archivos: desde los de la configuración del sistema, los de log, los de los usuarios, etc. En general, cada operativo utiliza su propio sistema de ficheros, caracterizándolo en muchos aspectos como pueden ser el rendimiento, la seguridad, la fiabilidad, etc. GNU/Linux es capaz de leer/escribir archivos con cualquiera de los sistemas de ficheros que actualmente existen, aunque para su propia raíz y directorios principales es necesario un sistema de ficheros que le permita ciertas operaciones. Generalmente, se suele utilizar el tipo ext2, ext3 o ReiserFS. El **ext2** es el más típico y extendido. Su rendimiento es bastante bueno, incorpora todo

tipo de mecanismos de seguridad y *tunning* y es muy fiable. **ext3** es la evolución del mismo, incorporando una tecnología llamada de *journaling*. Una de las principales ventajas de esta tecnología es que si hay un corte en el suministro de energía y el ordenador se apaga sin cerrarse adecuadamente, los sistemas de recuperación de ficheros son más efectivos. **ReiserFS** es un nuevo tipo de sistema que incorpora nuevas tecnologías de diseño que le permiten ser más rápido. En el proceso de instalación del sistema operativo se nos preguntará cuál de estos tres queremos usar. Generalmente se suele utilizar ext2 o ext3 por estar más probados que el ReiserFS.

Una característica muy importante de todos los sistemas operativos basados en UNIX es que todos los dispositivos del sistema se pueden tratar como si fueran ficheros. Igualmente, cuando queramos acceder al contenido de un CD, disquete o cualquier otro dispositivo de almacenamiento, deberemos **montarlo** en un directorio ya existente en el sistema y navegaremos por él como si se tratara de una carpeta más (el uso de diferentes unidades -A:,B:,C:,D:,. . . es un esquema existente únicamente en sistemas operativos tipo Windows^{TM}).

Lo primero que debemos tener claro es que todo el sistema de ficheros parte de una misma raíz, a la cual nos referiremos con el carácter "/ ". Es el origen de todo el sistema de ficheros y sólo existe una. Para organizar los ficheros adecuadamente, el sistema proporciona lo que llamaremos directorios (o carpetas), dentro de las cuales podemos poner archivos y más directorios. De este modo conseguimos una organización jerárquica como la que vemos en la siguiente figura:



Contenido complementario

El sistema de ficheros ext2 ha sido diseñado para manejar de forma muy rápida ficheros pequeños, que es lo que más suele tener un sistema operativo. Con el manejo y manipulación de grandes ficheros multimedia, no se desenvuelve tan bien, aunque siempre se puede hacer un poco de tunning para adaptarlo más a nuestras necesidades.

ANOTACIONES

2.3.2. Directorios del sistema

La mayoría de los sistemas operativos del mercado siguen el estándar FHS (http://www.pathname.com/fhs/), donde se especifican las principales características que debería tener cualquier sistema operativo. Entre ellas está la distribución en directorios que tenemos que hacer de nuestros archivos para tenerlos correctamente organizados y poder localizarlos de forma rápida y sencilla. En la mayoría de distribuciones basadas en GNU/Linux se siguen estas recomendaciones, encontrando los siguientes directorios principales:

- `/bin/`: comandos básicos para todos los usuarios del sistema.
- `/boot/`: archivos estáticos necesarios para el arranque del sistema.
- `/dev/`: dispositivos del sistema.
- `/etc/`: archivos de configuración del sistema y de las aplicaciones instaladas en el mismo.
- `/home/`: directorio para poner las carpetas *home* de los usuarios.
- `/lib/`: librerías esenciales para el núcleo del sistema y módulos del mismo.
- `/mnt/`: punto de montaje temporal para dispositivos.
- `/proc/`: procesos y variables del núcleo del sistema.
- `/root/`: directorio *home* para el *root* del sistema.
- `/sbin/`: comandos especiales para el *root* del sistema.
- `/tmp/`: archivos temporales. Según la distribución utilizada (o la configuración que utilicemos) se borran al arrancar el sistema o cada cierto período de tiempo.
- `/usr/`: segunda estructura jerárquica, utilizada para almacenar todo el software instalado en el sistema.

ANOTACIONES

- `/var/`: directorio para los *spoolers* de impresión, ficheros de log, etc.

Es muy recomendable conservar y no eliminar ninguno de estos directorios (o los que por defecto nos crea la distribución que utilizamos), ya que son básicos para el buen funcionamiento del sistema. Generalmente, los procesos de instalación de nuevas aplicaciones necesitan que exista la organización dada y muchos de los archivos de configuración de los programas deben estar en determinados directorios. Lo que sí que podemos hacer sin ningún tipo de restricción es crear nuevos directorios en la raíz del sistema o en cualquier otra carpeta.

2.3.3. Moviéndonos

Para movernos por la estructura de directorios debemos utilizar los comandos para listar contenidos y cambiar de carpeta. Cuando entramos en el sistema, es usual que el *login* nos sitúe en nuestro directorio *home*, que generalmente se suele referenciar con el carácter "~". Si queremos ver lo que hay en el directorio donde estamos situados, podemos listar los contenidos utilizando el comando `ls`. Debemos tener en cuenta que por defecto el comando no nos muestra los archivos que empiezan por un punto. Con el parámetro "`-a`" sí que nos mostraría absolutamente todos los ficheros. En todos los directorios existe una entrada "`.`" y otra "`..`". El punto es la referencia al directorio actual, mientras que los dos puntos seguidos hacen referencia al directorio inmediatamente superior (en el árbol de jerarquías) al actual. Naturalmente, cuando estamos situados en la raíz del sistema de ficheros, la entrada "`..`" no existirá porque nos encontramos en el nivel superior.

Contenido complementario

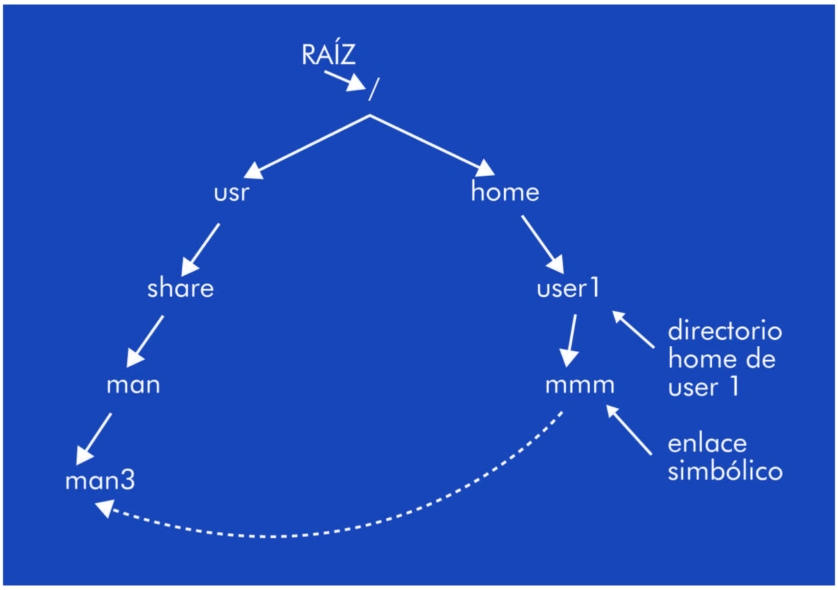
El hecho de que `ls` no nos muestre (por defecto) los archivos que empiezan por punto, es para que cada vez que listamos el contenido del directorio no tengamos que ver los ficheros y directorios de configuración de las aplicaciones que utilizamos (que suelen empezar por este carácter) y las entradas del directorio actual y anterior, que siempre existen.

Para cambiar de directorio podemos utilizar el comando `cd`. Si no le pasamos ningún parámetro, por defecto nos situará en nuestro directorio *home*. Generalmente, se le suele indicar dónde queremos ir, pasándolo de forma absoluta o relativa. De forma relativa significa que partiremos del directorio donde estamos en el momento de ejecutar el comando. Por ejemplo, si estamos en el directorio `/usr/bin/` y queremos ir al `/root/`, deberíamos introducir el siguiente comando: "`cd ../../root`" (los dos primeros puntos indican `/usr/` y los siguientes la raíz "`/`" del sistema, a partir de la cual ya podemos acceder a `/root/`). De forma absoluta siempre partimos de la raíz, de manera que el comando que utilizaríamos para el

ejemplo anterior sería: "`cd /root`". Para saber en qué directorio estamos, podemos utilizar el comando `pwd`.

2.3.4. Enlaces

Otros mecanismos que nos proporcionan la gran mayoría de sistemas de ficheros son los que llamamos *enlaces*. Un enlace es un puente a un archivo o directorio perteneciente al sistema; una referencia que podemos poner en cualquier sitio que nos interese y que actúa como un acceso directo a cualquier otro. Este mecanismo nos permite acceder a carpetas o archivos de forma más rápida y cómoda, sin tener que desplazarnos por la jerarquía de directorios. Vamos a verlo con un ejemplo: imaginemos que somos un usuario (user1) que necesita acceder frecuentemente al directorio `/usr/share/man/man3/`. En lugar de escribir el largo comando que nos situaría en el directorio en cuestión cada vez que necesitáramos desplazarnos a él, podemos crear un enlace en nuestro propio directorio que nos redireccione directamente hacia allí. El comando "`ln -s /usr/share/man/man3 mmm`" nos crearía este puente, que hemos llamado "`mmm`". El usuario sólo debería escribir (desde su directorio *home*) "`cd mmm`" y automáticamente el sistema lo redirigiría hacia `/usr/share/man/man3/`. Es importante tener en cuenta que al hacer un "`cd ..`" para ir al directorio superior, volveríamos al directorio *home* y no a `usr/share/man/`, ya que hemos accedido a él a partir de nuestro enlace. Podemos ver este esquema de forma gráfica en la siguiente figura:



Al crear el enlace del ejemplo anterior hemos pasado el parámetro "`-s`" al comando. Ello indica que queremos crear un **enlace simbólico**. Los enlaces simbólicos significan que sólo estamos creando un apuntador o puente hacia el fichero o directorio, de forma que si borrásemos el fichero destino, el enlace no apuntaría a ninguna parte. Si no ponemos el parámetro "`-s`" se crearía lo que llamamos un **enlace fuerte** (*hard link*) que, a diferencia del anterior, hace un duplicado del fichero. De hecho, internamente no es exactamente un duplicado, es como dos entradas que apuntan a los mismos datos. De este modo, si modificamos uno u otro, los dos quedan iguales. La ventaja de este tipo de enlace es que si borramos cualquiera de las dos copias del fichero la otra todavía se conserva. Este tipo de enlace no se utiliza demasiado porque complica la gestión y manipulación de los ficheros (siempre es mejor tener una sola copia de los archivos). Además, si hacemos un enlace fuerte de un directorio, todos los archivos y subdirectorios que contuviera también se deberían referenciar. Por esta razón sólo el *root* del sistema puede hacer enlaces fuertes de directorios. Otra diferencia es que con un enlace simbólico podemos ver a qué fichero estamos apuntando, mientras que con uno fuerte no podemos (debido al mecanismo que se utiliza internamente para ellos).

Contenido complementario

Un enlace fuerte sólo se puede crear entre ficheros o directorios de una misma unidad debido al mecanismo interno que se utiliza para gestionarlos.

2.3.5. Permisos

En cualquier sistema operativo multiusuario necesitamos que los ficheros que guardamos en nuestro disco puedan tener una serie de propiedades que nos permitan verlos, modificarlos o ejecutarlos para los usuarios que nosotros definamos. Aunque hay varias alternativas para hacer esto, GNU/Linux utiliza el sistema clásico de UNIX, que, combinado con todos los mecanismos de gestión de usuarios y grupos, nos permite cualquier configuración posible. Lo que interesa es definir, para cada fichero o directorio, a qué usuario y grupo pertenece y qué permisos tiene para cada uno de ellos, así como para el resto de usuarios del sistema. Ejecutando "`ls -l`" veremos cómo en cada archivo del directorio donde estamos aparece una línea parecida a la siguiente:

```
-rwxr-xr-x 1 user1 grupo1 128931 Feb 19 2000 gpl.txt
```

ANOTACIONES

Los primeros diez caracteres (empezando por la izquierda) nos indican los permisos del fichero de la siguiente manera:

- Carácter 1: esta entrada nos indica si es un fichero o un directorio. En caso de ser un fichero, aparece el carácter "–", mientras que por los directorios aparece una "`d`".

- Caracteres 2, 3, 4: nos indican, respectivamente, los permisos de lectura, escritura y ejecución para el propietario del fichero. En caso de no tener el permiso correspondiente activado, encontramos el carácter "–" y si no "`r`", "`w`" o "`x`", según si lo podemos leer (*Read*), escribir (*Write*) o ejecutar (*eXecute*). En el tercer carácter, además, podemos encontrarnos una "`s`", que nos indica si el archivo es de tipo SetUserId, que quiere decir que al ejecutarlo obtendrá los permisos del propietario del fichero. Si sólo tiene el permiso "`x`", cuando el programa se ejecuta lo hace con los permisos de quien lo haya lanzado.

- Caracteres 5, 6, 7: estos caracteres tienen exactamente el mismo significado que anteriormente, pero hacen referencia a los permisos concedidos a los usuarios del grupo al que pertenece el fichero.

- Caracteres 8, 9, 10: igual que en el caso anterior, pero para los otros usuarios del sistema.

Contenido complementario

El mecanismo de SetUserId es muy útil cuando un programa necesita tener los permisos de su propietario para acceder a ciertos archivos o hacer algún tipo de operación en el sistema. De todos modos, debemos vigilar mucho con este tipo de ficheros porque pueden suponer fallos de seguridad en el sistema si son mal utilizados.

Después de estos 10 caracteres encontramos una cifra que nos indica el número de enlaces fuertes que tiene el fichero. Para los directorios, este número indica cuántas carpetas hay dentro de él además de los enlaces fuertes que tiene (cuando no hay ninguno, el número es 2, debido a la gestión interna del operativo). A continuación vemos el propietario y el grupo del archivo, seguido del tamaño (en *bytes*) que ocupa y la fecha de la ultima modificación. En todos los ficheros se guarda su fecha de creación, último acceso y modificación, que podemos manipular con el comando `touch`. Al final hay el nombre del fichero, dónde se diferencian minúsculas de mayúsculas y podemos tener todo tipo de caracteres sin ningún problema.

Para cambiar los permisos de un determinado archivo podemos utilizar el comando `chmod`. Debemos tener en cuenta que sólo el propietario del archivo (o el *root*) puede cambiar estos permisos, ya que

ANOTACIONES

si no, el mecanismo no tendría ningún sentido. Podemos utilizar este comando de muchas maneras diferentes, pero las dos más frecuentes son las siguientes:

- El primer modo de utilizarlo es del estilo "`chmod XXX nombreArchivo`". Las "`X`" deben ser tres números entre 0 y 7. El primer número indica los permisos que queremos establecer para el usuario, el segundo, para el grupo y el tercero, para el resto. Para interpretar correctamente los permisos que daremos utilizando los números del 0 al 7, debemos hacer uso de la representación binaria del número en cuestión, de forma que el primer dígito indicará el permiso de escritura, el segundo, el de lectura y el tercero, el de ejecución. En cada caso un 0 indica que no se da el permiso en cuestión y el 1 indica que sí que se da. En la siguiente tabla podemos ver esta relación:

Representación decimal	Representación binaria	Significado
0	000	`---`
1	001	`--x`
2	010	`-w-`
3	011	`-wx`
4	100	`r--`
5	101	`r-x`
6	110	`rw-`
7	111	`rwx`

- El otro modo de utilizar el comando es indicando de forma explícita qué permiso queremos dar o eliminar del fichero. La manera de hacerlo es indicando, primero, si nos referimos a los permisos del usuario, grupo o al resto con las letras "`u`", "`g`" u "`o`" respectivamente. Seguidamente, debemos añadir un "+" o "–" según si queremos añadir o eliminar el atributo, que indicaremos con "`r`", "`w`", "`x`" o "`s`" (este último para el SetUserId). Además, podemos hacer todas las combinaciones posibles, refiriéndonos a más de un permiso y/o usuarios. Por ejemplo, "`chmod go+r gpl.txt`"daría el permiso de lectura al grupo y a los otros usuarios para el fichero `gpl.txt`.

ANOTACIONES

Contenido complementario

Si se permitiera a los usuarios cambiar el propietario de sus ficheros, la seguridad del sistema quedaría comprometida, porque se podrían realizar acciones maliciosas y después cambiar el propietario de los archivos utilizados inculpando a otros usuarios.

Para cambiar el propietario de un fichero existe el comando `chown`, que sólo puede utilizar el *root* por razones de seguridad. Para cambiar el grupo de un determinado archivo, se puede utilizar el comando `chgrp`. Como podemos suponer, cuando un usuario crea un nuevo archivo, el sistema pone como propietario al usuario que lo ha creado y lo da como perteneciente al grupo primario del mismo usuario. Los permisos que se ponen por defecto al crear un nuevo archivo los podemos configurar con el comando `umask`, al que debemos pasar la misma notación de tres números decimales entre 0 y 7 que veíamos anteriormente pero complementados. Por ejemplo, si queremos que nuestros ficheros se inicialicen con los permisos "`rw-r--r--`", deberíamos escribir "`umask 133`".

2.3.6. Manipulación, patrones y búsquedas

Contenido complementario

La sintaxis de los *patterns* puede llegar a ser muy compleja, permitiéndonos referenciar cualquier conjunto de archivos que queramos.

Ahora que ya sabemos movernos correctamente por la jerarquía de directorios, también necesitamos saber cómo copiar, eliminar y manipular correctamente otros aspectos de los ficheros. El comando `rm` es el que se encarga de eliminar los archivos que le indiquemos. Para eliminar un directorio, podemos utilizar el comando `rmdir`, aunque sólo lo borrará cuando éste esté vacío (si quisiéramos borrar completamente un directorio y todo su contenido, podríamos utilizar "`rm -r`"). Para copiar archivos de un lugar a otro tenemos el comando `cp`, al que siempre debemos indicar el fichero o directorio origen y el lugar o nombre de destino, aunque sea en el directorio actual. De este modo, si queremos copiar el archivo `/home/user1/gpl.txt` en el directorio actual (y con el mismo nombre) deberíamos escribir "`cp /home/user1/gpl.txt .`". Si en lugar de copiar los archivos queremos moverlos de sitio, podemos utilizar el comando `mv`.

Un mecanismo muy útil que nos proporciona el sistema son los ***patterns*** ('patrones'). Hasta ahora hemos visto cómo aplicar ciertas operaciones sobre un determinado archivo. Cuando estamos manipulando un sistema, en muchos casos nos interesará aplicar alguna de las operaciones que hemos visto pero sobre un grupo grande de ficheros. Los patrones nos permitirán aplicar las operaciones que queramos especificando en una sola instrucción varios ficheros que cumplan con una serie de características concretas. Debemos verlos como plantillas de nombres, de manera que el carácter "*" significa cualquier cadena de caracteres po-

ANOTACIONES

sibles y el "?" nos sirve como comodín a cualquier carácter. De este modo, si queremos listar todos los archivos que empiecen por "s", que después tengan cualquier otro carácter, les siga una "a", y después cualquier otra cadena, podríamos utilizar "`ls s?a*`". Entre "`[]`" podemos incluir otros caracteres, indicando que el patrón tiene éxito si se encuentra alguno de ellos en el nombre. Por ejemplo, si quisiéramos referenciar todos los archivos que empiecen por "a" o por "b" y que continúan con cualquier otra cadena, podríamos escribir el *pattern* "`[ab]*`". Si después de "`[`" pusiéramos el carácter "`!`" ("`[!ab]*`") indicaríamos que el *pattern* coincide con cualquier archivo que no empiece por "a" o "b". Finalmente, para facilitar ciertas búsquedas, dentro de "`[]`" podemos especificar clases de caracteres de la siguiente manera: "`[:clase:]`", donde la "`clase`" puede ser cualquiera de las nombradas en la siguiente tabla:

clase	significado	clase	significado
alnum	[A-Za-z0-9]	alpha	[A-Za-z]
blank	[\]	cntrl	cars de control
digit	[0-9A-Fa-f]	graph	cars imprimibles (sin espacios)
lower	[a-z]	print	cars imprimibles (con espacios)
punct	[.,¡!¿?:;] . . .	space	[ ]
upper	[A-Z]	xdigit	[0-9A-Fa-f]

A-Z indica caracteres de la A a la Z, \t es el tabulador y \n es un salto de línea.

Naturalmente, los *patterns* los podemos utilizar con cualquiera de los comandos que hemos visto y la mayoría de los que veremos a continuación. Además, la mayor parte de los comandos de listado, eliminación, copia, etc. de ficheros también permiten que se les pase un parámetro (generalmente "`-r`") para realizar las acciones respectivas de forma recursiva. De este modo, se irá entrando y ejecutando la instrucción correspondiente en todos los archivos y directorios, a partir de donde nos encontramos y hasta llegar al último nivel de la jerarquía.

Otro tipo de operación muy útil es la búsqueda de ficheros. Tenemos varios comandos que nos permiten realizar búsquedas de diferentes tipos sobre todos los ficheros del sistema.

Nota

Si queremos actualizar la base de datos interna que utiliza el comando `locate`, podemos utilizar updatedb.

`find`	Es el comando más versátil para realizar esta acción. Nos permite filtrar los ficheros para encontrar desde los que tienen un determinado nombre, los modificados o creados a partir de una cierta fecha, los que tienen ciertos permisos, etc. Su única desventaja es que no utiliza ningún tipo de mecanismo para acelerar la búsqueda, con lo cual, éstas pueden tardar bastante.
`locate`	Se trata de otro comando, pero, a diferencia del anterior, utiliza una base de datos interna que se actualiza periódicamente y nos permite hacer búsquedas bastante más rápidas. Debemos tener en cuenta, sin embargo, que los resultados no siempre estarán actualizados, además de que no podemos realizar búsquedas tan versátiles como con `find`.
`whereis`	Por último, `whereis` está orientado a la búsqueda de los archivos binarios (los ejecutables), de ayuda o los de código fuente de un determinado programa.

2.3.7. Tipos y contenido de ficheros

Contenido complementario

Utilizar la extensión para determinar el tipo de un archivo no es un sistema muy eficaz, ya que cualquiera puede cambiarla y generar confusiones y errores en el sistema.

Los archivos que tenemos en nuestro sistema pueden ser de muchos tipos diferentes: ejecutables, de texto, de datos, etc. A diferencia de otros sistemas que utilizan la extensión del archivo para determinar de qué tipo son, GNU/Linux utiliza un sistema denominado de **magic numbers**, determinando con un número mágico el tipo de fichero según sus datos (se pasan una serie de tests que intentan determinar de qué tipo es el fichero). El comando `file` nos lo indica.

Si necesitamos ver el contenido de un fichero, uno de los comandos básicos es el `cat`. Pasándole el nombre o nombres de los archivos que queremos ver, se muestra por pantalla. Debemos intentar no mostrar ficheros ejecutables o de datos por pantalla, ya que el volcado de caracteres no imprimibles nos dejaría la consola con caracteres no comprensibles (siempre la podemos reiniciar tecleando `reset` o `tset`). Para ficheros muy extensos, nos irán mucho mejor los comandos `less` o `more`, que permiten desplazarnos por el fichero de forma progresiva. Si el tipo de fichero es binario y queremos ver qué contiene, podemos utilizar los comandos `hexdump` u `od` para ver el contenido de forma hexadecimal u otras representaciones. `strings` nos buscará las cadenas de caracteres dentro de un fichero binario y las mostrará por pantalla.

Otro tipo de comandos muy útiles son los que nos buscan un cierto patrón en el contenido de los ficheros. Con el comando `grep` le podemos pasar como segundo parámetro el nombre del archivo y

ANOTACIONES

como primero el *pattern* que queramos buscar (con la sintaxis que veíamos anteriormente, extendida a otras opciones). Además, el comando nos permite otras múltiples acciones, como contar el número de líneas donde aparece el patrón (parámetro "-c"), etc. Con `cut` podemos separar en campos el contenido de cada línea del fichero especificando qué carácter es el separador, muy útil en tareas de administración del sistema para su automatización. También podemos coger un determinado número de líneas del principio o fin de un archivo con los comandos `head` y `tail` respectivamente. Con `wc` podemos contar el número de líneas o palabras, la máxima longitud de línea de un fichero, etc.

Finalmente, para acabar con esta sección de manipulación de ficheros, lo único que nos falta por ver es cómo comparar diferentes archivos. Igual que con las otras operaciones, tenemos varios comandos que nos permiten hacerlo. `diff`, `cmp` y `comm` realizan comparaciones de diferentes formas y métodos en los ficheros que indicamos. `sdiff`, además, permite mezclarlos a nuestra elección.

2.4. Los procesos

El hecho de que el sistema operativo sea multitarea implica que podemos lanzar más de un programa a la vez. Un proceso no es más que un programa o aplicación que se encuentra cargado en la memoria y en proceso de ejecución. Aunque nuestro ordenador sólo disponga de una CPU, el sistema operativo se encarga de repartir el tiempo de procesamiento de la misma para que varios procesos puedan ir realizando sus operaciones, dando la sensación de que se están ejecutando todos a la vez.

Contenido complementario

La gestión de procesos es un aspecto vital en todo sistema operativo, ya que determina el tiempo de respuesta de nuestras aplicaciones, la eficiencia con que se utiliza la memoria y la CPU, etc.

Para identificar de forma inequívoca cada proceso, el núcleo del sistema les asigna un número llamado PID (*Process IDentification*). Aunque podríamos pensar que con solo el nombre ya los podríamos referenciar, es imprescindible disponer de este número porque podemos ejecutar un mismo programa tantas veces como queramos, al mismo tiempo que se ejecutan diferentes instancias del mismo. Para saber qué procesos se están ejecutando, podemos utilizar el comando `ps`. Para explorar un poco más todo este mecanismo de pro-

ANOTACIONES

cesos, explicaremos con más detalle algunos de los parámetros que le podemos pasar a este comando:

- "`T`": esta opción viene por defecto y nos indica que sólo se mostrarán los procesos que se están ejecutando en el terminal dónde nos encontramos o que se hayan lanzando a partir de él.
- "`-a`": nos muestra los procesos de todos los terminales del sistema.
- "`-A`": nos muestra todos los procesos del sistema. Si ejecutamos el comando, veremos que, aparte de los programas que los usuarios ejecutan, hay otros. Muchos de ellos ejecutan las funciones necesarias para que el operativo funcione correctamente, otros son los servidores de aplicaciones configurados, etc.
- "`-l`": enseña información extendida para cada proceso, como el tiempo de CPU que ha utilizado, el terminal donde se ejecuta, etc. En la segunda columna también podemos ver el estado del proceso. Aunque el sistema tenga muchos procesos ejecutándose en un mismo instante de tiempo, ello no implica que todos necesiten tiempo de CPU constantemente. Por ejemplo, cuando un servidor de páginas web no tiene ninguna petición, no es necesario que haga absolutamente ninguna operación. Aunque esté en memoria preparado para ejecutarse al recibir una petición, es mejor que no pase en ningún momento por la CPU, ya que ésta puede utilizarse para otros procesos que sí que la necesitan. Internamente, el sistema operativo tiene implementados una serie de mecanismos muy eficaces para gestionar toda esta clase de operaciones. De este modo, un proceso puede estar en los siguientes estados (mostrados con el carácter correspondiente):

– "`D`": proceso ininterrumpible. Este tipo de proceso generalmente suele pertenecer a la entrada/salida de algún dispositivo que se dañaría si dejara de ser atendido.

– "`R`": proceso que en el momento de ejecutar el comando también se está ejecutando, o sea, todos aquellos que están en **cola de ejecución**. La cola de ejecución de procesos es donde se ponen todos aquellos que se van repartiendo el tiempo de la CPU.

ANOTACIONES

– "`S`": proceso dormido o esperando que ocurra algún tipo de evento para que el sistema lo despierte y lo ponga en la cola de ejecución.

– "`T`": proceso que ha sido detenido por el usuario o el sistema.

– "`Z`": proceso *zombie*. Este estado indica que el proceso ha tenido algún fallo y no funciona correctamente. Generalmente es mejor eliminar este tipo de procesos.

Otro comando muy útil es el `top`, que nos informa de forma interactiva de los procesos del sistema, del estado de utilización de la CPU, la memoria utilizada y libre, la RAM que utiliza cada proceso, etc. Este programa es muy indicado cuando el sistema no responde adecuadamente o notamos alguna disfunción extraña, ya que nos permite localizar rápidamente qué proceso está afectando negativamente al rendimiento del sistema.

Contenido complementario

Con los comandos de manipulación de procesos podemos realizar cualquier acción que nos interese: desde pausar los procesos de un usuario concreto, eliminar aquellos que no nos interesan o hacer que algunos ocupen más tiempo la CPU para que vayan más rápido.

Como vemos, el sistema nos informa sobre todos los aspectos posibles de los procesos del sistema. Además de esto, podemos enviar ciertas señales a los procesos para informarles de algún evento, podemos sacarlos de la cola de ejecución, eliminarlos, darles más prioridad, etc. Saber manipular correctamente todos estos aspectos también es muy importante, ya que nos permitirá utilizar nuestro ordenador de forma más eficiente. Por ejemplo, si somos administradores de un centro de cálculo donde la mayoría de aplicaciones que se ejecutan necesitan mucho tiempo de CPU, podríamos configurar el sistema para hacer que los más urgentes se ejecuten con más prioridad que otros y acaben primero. El comando `kill` nos permite enviar **señales** a los procesos que nos interese. En general, todos los programas se diseñan para que puedan recibir este tipo de señales. De este modo, según el tipo de señal recibido saben que deben realizar unas operaciones u otras. Hay muchos tipos diferentes de señales, que podemos ver en el manual de `kill`, aunque las más utilizadas son las que nos sirven para obligar a un proceso a que termine o pause su ejecución. Con la señal `TERM` ("`kill -15 PID`"), le indicamos al proceso que queremos que termine, de modo que al recibir la señal deberá guardar todo lo necesario y acabar su ejecución. Si hay algún tipo de problema o el programa no está preparado para recibir este tipo de señal, podemos utilizar `KILL` ("`kill -`

Contenido complementario

Para tratar las señales en un *shell script* (véase más adelante cómo programarlos), podemos utilizar el comando `trap`.

ANOTACIONES

9 PID"), que automáticamente lo expulsa de la cola de ejecución. `killall` sirve para referirnos al nombre de varios procesos a la vez en lugar de referenciarlos por su PID y, de esta forma, enviarles una señal a todos a la vez. Con el comando `skill` también podemos enviar señales a los procesos, pero con una sintaxis diferente. Por ejemplo, si queremos detener todas las ejecuciones de un determinado usuario, podríamos utilizar "`skill -STOP -u nombreLogin`", con lo que todos los procesos de dicho usuario se pararían. Para reiniciarlos de nuevo, podríamos pasar la señal de CONT. Cuando estamos ejecutando algún programa en una consola y queremos pasarle la señal de `TERM`, podemos utilizar la combinación de teclas CTRL+C. Con CTRL+Z podemos pausar un programa y revivirlo con `fg`.

Otra manera de ver los procesos es por su jerarquía. Igual que en el sistema de ficheros, los procesos siguen una cierta jerarquía de padres a hijos. Todo proceso debe ser lanzado a partir de otro, sea el propio intérprete de comandos, el entorno gráfico, etc., de manera que se crea una relación de padres a hijos. Con el comando `pstree` podemos ver esta jerarquía de forma gráfica. Si lo ejecutamos, veremos cómo el padre de todos los procesos es uno llamado `init`. A partir de éste parten todos los demás, que a la vez pueden tener más hijos. Esta estructura jerárquica es muy útil, ya que, por ejemplo, matando a un proceso padre que contiene muchos otros hijos, también matamos a todos sus hijos. También nos puede servir para identificar de dónde parten ciertos procesos, etc. Si no le pasamos ningún parámetro al comando, por defecto compacta todos los procesos con un mismo nombre para no mostrar una estructura demasiado grande, aunque esto también es configurable a partir de sus parámetros.

Todos los procesos del sistema tienen una cierta prioridad. Como decíamos antes, esta prioridad indica el tiempo de CPU que se le dejará al proceso. Cuanto más prioritario sea el proceso, más tiempo de ejecución tendrá respecto a los otros. El rango de prioridades va desde el –20 al 19, de mayor a menor. Para lanzar un proceso con una determinada prioridad, podemos utilizar el comando `nice`. Si queremos dar una prioridad diferente a un proceso que ya esté en ejecución, podemos utilizar `renice`. Sólo el *root* puede utilizar el rango de prioridades negativas; así, el sistema se asegura de que el *root* cuente siempre con la posibilidad de ejecutar procesos más rápidamente que los usuarios. Por defecto, la prioridad con que se ejecutan

ANOTACIONES

los programas es la 0. Un aspecto que habrá que considerar es que con todo este mecanismo de prioridades no podemos medir el tiempo de ejecución real de un proceso porque la CPU se reparte entre todos los que tengamos en la cola de ejecución. En centros de cálculo donde se factura según el tiempo de utilización de las máquinas, es muy importante poder medir adecuadamente este aspecto. Por este motivo, el sistema nos proporciona el comando `time`, el cual, al pasarle el programa que queremos medir, nos devuelve el tiempo real de CPU que ha utilizado.

2.5. Otros comandos útiles

2.5.1. La ayuda del sistema

Como hemos dicho a lo largo del documento, todos los comandos tienen multitud de opciones y parámetros diferentes que nos permiten manipularlos a nuestra elección. Desde el principio se tuvo muy en cuenta que es imprescindible contar con una buena documentación para todos ellos. Igualmente, toda esta información es necesaria para los ficheros de configuración del sistema, las nuevas aplicaciones que utilizamos, etc. Por ello, el mismo sistema incorpora un mecanismo de manuales con el que podemos consultar casi todos los aspectos de los programas, utilidades, comandos y configuraciones existentes. El comando más utilizado es el `man`, que nos enseña el manual del programa que le indicamos como parámetro. Por defecto, esta documentación se muestra por medio del programa `less`, con el cual podemos desplazarnos hacia delante y hacia atrás con las teclas de AVPÁG y REPÁG, buscar una palabra con el carácter "`/`" seguido de la palabra ("`n`" nos sirve para buscar las siguientes ocurrencias y "`N`" para las anteriores), "`q`" para salir, etc. Los manuales del sistema están divididos en diferentes secciones según la naturaleza de los mismos:

1) Programas ejecutables (aplicaciones, comandos, etc.).

2) Llamadas al sistema proporcionadas por el *shell*.

3) Llamadas a librerías del sistema.

ANOTACIONES

4) Archivos especiales (generalmente los de dispositivo).

5) Formato de los archivos de configuración.

6) Juegos.

7) Paquetes de macro.

8) Comandos de administración del sistema (generalmente aquellos que sólo el *root* puede utilizar)

9) Rutinas del núcleo.

Contenido complementario

Para realizar sus búsquedas de forma rápida, la aplicación man utiliza una base de datos interna que va a buscar por los archivos que contienen los manuales y los indexa de forma adecuada. Si queremos actualizar este manual (aunque normalmente el mismo sistema ya lo hace automáticamente), podemos utilizar el comando `mandb`.

Si hay más de un manual disponible para una misma palabra, podemos especificarlo pasándole el número correspondiente de la sección deseada antes de la palabra, por ejemplo "`man 3 printf`". Como los otros comandos, `man` tiene multitud de opciones diferentes documentadas en su propio manual ("`man man`"), a partir de las cuales podemos hacer búsquedas automáticas, crear un fichero del manual en formato imprimible, etc. Una de estas opciones, que nos puede ir muy bien en las ocasiones que no sepamos exactamente el programa que estamos buscando, es "`-k`" (el comando `apropos` hace casi exactamente lo mismo). Con "`man -k`" seguido de una palabra que haga referencia a la acción que queramos realizar se buscará por entre todos los manuales del sistema y se mostrarán los que en su descripción o nombre aparezca la palabra indicada. Así, podemos encontrar lo que queremos sin tener que recurrir a ningún libro o referencia externa al sistema.

Si el manual no nos proporciona toda la información que necesitamos, podemos usar el comando `info`, que es lo mismo que el manual pero aún más extendido. Si lo único que queremos es tener una breve referencia de lo que hace un determinado programa/librería/etc., podemos utilizar el comando `whatis`.

2.5.2. Empaquetado y compresión

Comprimir un archivo, agrupar varios en uno solo o ver qué contiene un archivo comprimido son tareas que efectuaremos frecuentemente para hacer copias de seguridad, transportar archivos de un sitio a otro, etc. Aunque existen multitud de programas diferentes que nos permiten llevar a cabo esta clase de operaciones, ge-

ANOTACIONES

neralmente en todos los sistemas GNU/Linux encontraremos la herramienta `tar`. Este programa nos permite manipular de cualquier manera uno o varios archivos para comprimirlos, agruparlos, etc. Aunque sus múltiples opciones son inacabables y tiene muchísima flexibilidad, aquí sólo explicaremos algunas de las más básicas para hacernos una idea de lo que podemos hacer con él. La sintaxis que utiliza es la siguiente: "`tar opciones archivoDestino archivosOrigen`", donde el archivo de destino será el nuevo fichero que queremos crear y los de origen serán los que se agruparán o comprimirán. Es importante tener en cuenta que si queremos agrupar toda una carpeta, por defecto el proceso es recursivo, de forma que al empaquetarla ésta recorrerá todos sus niveles y agrupará todo lo que contenga. Para crear un nuevo archivo, debemos pasarle el parámetro "`c`", y si lo queremos guardar en un archivo, debemos pasarle el "`f`". De este modo, "`tar cf final.tar o*`" empaquetará todos los archivos del directorio actual que empiecen por "`o`". Si además quisiéramos comprimirlos, podríamos utilizar "`czf`" con lo que se utilizaría el programa `gzip` después de empaquetarlos. Para desempaquetar un determinado archivo, el parámetro necesario es el "`x`", de modo que deberíamos escribir "`tar xf`" indicando el fichero empaquetado. Si estuviera comprimido, deberíamos pasar "`xzf`".

Aunque con el mismo `tar` podemos comprimir archivos, la aplicación en sí misma no es de compresión. Como hemos dicho, para ello utiliza programas externos como el `gzip`. El `gzip` utiliza un formato de compresión propio y diferente del tan popularizado `zip`, que también podemos utilizar instalando la aplicación correspondiente. Otra aplicación de compresión bastante utilizada y que proporciona muy buenos resultados es el `bzip2`. En la siguiente tabla podemos ver la extensión que se suele utilizar para identificar qué formato utiliza un archivo comprimido o empaquetado:

extensión	formato
"`.tar`"	tar
"`.gz`"	gzip
"`.tgz`"	tar + gzip
"`.bz2`"	bzip2
".zip"	zip
".z"	compress

ANOTACIONES

2.5.3. Operaciones de disco

La gestión y manipulación de los discos duros del ordenador es otro aspecto fundamental en las tareas de administración del sistema. Aunque más adelante veremos cómo configurar adecuadamente los discos que tengamos instalados en el ordenador, en esta sección explicaremos cuáles son los comandos necesarios para ver información relativa a los mismos. Todo disco duro está dividido en **particiones**, a las que podemos acceder como si de un dispositivo independiente se tratara, y denominaremos **unidad**. Esto es muy útil porque nos permite separar de forma adecuada la información que tengamos en el sistema, para tener más de un sistema operativo instalado en el mismo disco, etc. El comando `df` nos muestra, de cada unidad montada en el sistema, el espacio que se ha utilizado y el que está libre. Vamos a interpretar la siguiente salida de `df`:

Contenido complementario

El tamaño del bloque y otros muchos parámetros se pueden configurar al formatear una partición del disco duro (con el sistema ext2 o ext3). Estos parámetros se pueden ajustar para hacer que el sistema se adapte mejor a nuestras necesidades y conseguir mayor eficiencia.

Filesystem	1k-blocks	Used	Available	Use%	Mounted on
/dev/hda1	7787712	421288	6970828	6%	/
/dev/hdb1	19541504	5742384	13799120	29%	/info
/dev/hdc	664432	664432	0	100%	/CD-ROM

Como podemos ver, por cada partición o dispositivo montado en el sistema el comando nos muestra la cantidad de bloques disponibles y utilizados. El bloque de disco es una unidad que se utiliza internamente en los dispositivos de almacenamiento para que el manejo de los mismos sea más efectivo. Por defecto, este comando nos enseña la información por bloques de 1k, aunque pasándole el parámetro "`-h`" (*human readable*) lo podríamos ver de forma más amena. La primera línea siempre nos muestra la raíz del sistema de ficheros (el ***root filesystem***) y después los otros dispositivos. Fijémonos como también nos muestra su punto de anclaje (en la última columna), que es la carpeta donde deberíamos ir para poder ver su contenido.

Otro comando muy útil es `du`, que nos muestra realmente lo que nos ocupa un fichero en disco. Para entender claramente qué queremos decir con esto, debemos profundizar un poco más en la organización interna de los discos y en cómo el sistema operativo los manipula. Tal como decíamos anteriormente, por razones de eficiencia el sistema operativo divide el espacio del disco en pequeños trozos llama-

ANOTACIONES

dos *bloques*. El tamaño del bloque es configurable y generalmente depende del tamaño del disco, aunque también lo podemos configurar para adaptarlo mejor a nuestras necesidades. Cada vez que queremos añadir un nuevo archivo, el sistema operativo le asigna un bloque. De este modo, al leer u operar sobre él, el operativo puede leer directamente todo un bloque (del tamaño configurado) en un solo paso. Cuando el fichero ocupa más de un bloque, se le asignan más, intentando que queden lo más juntos posible, de modo que se puedan leer consecutivamente e incrementando, así, la velocidad de lectura. El único inconveniente de este sistema es el desaprovechamiento que se hace de los bloques cuando los ficheros son muy pequeños, ya que si un determinado archivo no ocupa todo el bloque, el espacio restante no se puede aprovechar para ningún otro. De todos modos, este tipo de organización es el que utilizan todos los sistemas de ficheros existentes, ya que es lo más rentable para aprovechar el disco duro. El comando `du`, pues, nos muestra el número de bloques que realmente utiliza un determinado archivo en el disco.

Para saber los parámetros que tenemos configurados en nuestras unidades de disco formateadas con ext2 o ext3, podemos utilizar el comando `dumpe2fs`, pasándole la partición concreta. Veremos cómo hay multitud de opciones diferentes que nos permiten ajustar muy bien el comportamiento del mismo (en el manual encontraremos qué significa cada opción). De todos modos, una vez hayamos formateado una partición, ya no podremos modificar casi ninguna de estas opciones. Si quisiéramos cambiarlas, deberíamos copiar toda la información de la partición, formatear de nuevo y volver a copiar los archivos originales.

Contenido complementario

La desfragmentación de un disco no es más que la reorganización de los bloques de los ficheros para que queden en lugares consecutivos y su acceso sea más rápido. En los sistemas de ficheros que utilizamos con GNU/Linux no hace falta desfragmentar los discos (aunque hay programas al efecto) porque el sistema se encarga automáticamente de su buena organización.

Las funciones del núcleo que se encargan de la gestión de ficheros utilizan una serie de métodos para agilizar los procesos de lectura y escritura de los mismos. Uno de ellos es la utilización de una caché de disco, de forma que no se haya de estar constantemente leyendo y escribiendo en el dispositivo físico, que resulta un proceso lento y costoso. Lo único que hace el mecanismo de caché es mantener una copia del fichero con el que se está trabajando en la memoria RAM (mucho más rápida), de modo que el proceso sea transparente para el usuario (la copia a disco se realiza según algún tipo de política implementada en el núcleo). El único problema de esta gestión es que si tenemos un corte en la alimentación y no hemos cerrado correcta-

mente el sistema, es posible que algunos ficheros no se hayan podido guardar en el disco físico y tengamos alguna **inconsistencia** en el sistema de ficheros. El programa `fsck` comprueba y arregla un sistema de ficheros que haya quedado en este estado. Aunque lo podemos ejecutar cuando queramos, generalmente el mismo sistema operativo lo ejecuta cuando en el proceso de arranque detecta que el sistema no se cerró adecuadamente (antes de apagar el ordenador, debemos ejecutar el comando `shutdown`, que se encarga de lanzar todos los procesos necesarios para que los programas acaben, se desmonte el sistema de ficheros, etc.). En este sentido, el sistema de ficheros ext3 es más eficaz que su predecesor, ya que el ***journaling*** le permite recuperar más información de los ficheros perdidos y más rápidamente.

Naturalmente, si los ficheros que tratamos en nuestro sistema son muy críticos y no podemos, en ningún caso, permitirnos el hecho de perderlos, también podemos configurar el operativo para que no utilice el sistema de caché de disco. De todos modos, es muy recomendable utilizar este mecanismo porque incrementa mucho el rendimiento del sistema. Si en algún momento nos interesa sincronizar la información de la caché de disco con el disco físico, podemos utilizar el comando `sync`. Finalmente, también podemos comprobar la integridad física de una partición utilizando el comando `badblocks`, que lleva a cabo un chequeo sobre el dispositivo indicado para comprobar que no haya ninguna zona dañada.

La mayoría de los comandos expuestos en esta sección necesitan de permisos especiales para ejecutarse, de forma que sólo el *root* podrá utilizarlos.

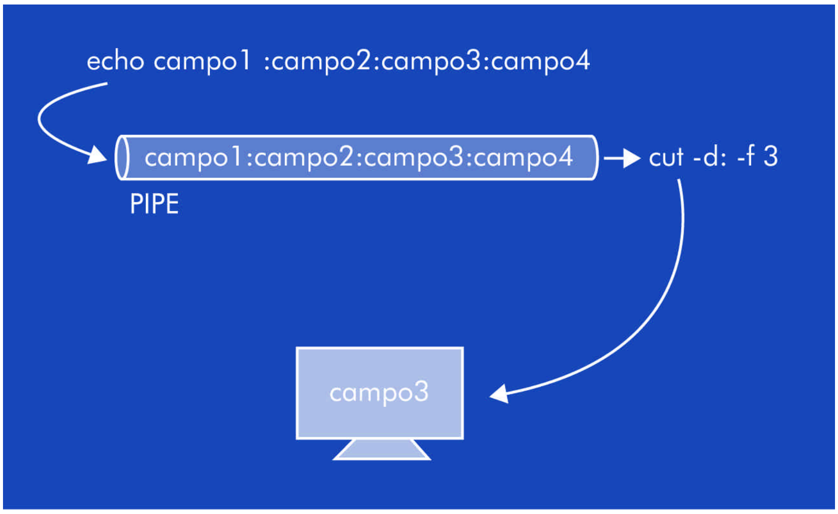
2.6. Operaciones con comandos

2.6.1. Redireccionamientos

Una vez hemos aprendido a utilizar algunos de los comandos del sistema, es muy probable que en algunos casos nos interese utilizarlos de forma simultánea para agilizar las acciones que queremos realizar. Una operación muy interesante consiste en poder coger la salida

ANOTACIONES

de un comando para que sirva de entrada a otro y procesarla adecuadamente. El sistema operativo utiliza un mecanismo de ***pipes*** (tuberías), que nos permite redirigir las salidas de cualquier comando o programa hacia donde queramos. Su funcionamiento es muy simple: se trata de poner el carácter "`|`" entre los comandos, de manera que la salida del primero sirve como entrada para el segundo. Vamos a verlo con un ejemplo: al escribir el comando "`echo campo1:campo2:campo3:campo4`", lo único que conseguiríamos sería que por pantalla nos apareciera "`campo1:campo2:campo3:campo4`". Si de esta salida sólo quisiéramos coger el "`campo3`", podríamos redirigirla con un *pipe* hacia el comando `cut`, para que seleccione únicamente el campo que nos interesa de la siguiente manera: "`echo campo1:campo2:campo3:campo4 | cut -d: -f 3`". En la siguiente figura podemos ver este ejemplo de forma gráfica:



Naturalmente, podemos conectar tantos *pipes* como necesitemos para realizar acciones más prácticas que la que acabamos de ver. Otro tipo de redireccionamientos muy prácticos son aquellos que están relacionados con los ficheros. Este tipo de redireccionamiento nos permite coger toda la salida de un comando o programa y guardarla en un fichero utilizando el carácter ">", igual que hacíamos con "|". Por ejemplo, si queremos guardar en un nuevo fichero todo lo que vayamos escribiendo hasta apretar CTRL+C, podríamos utilizar lo siguiente: "`cat > prueba.txt`". Con ">>" podemos hacer exactamente lo mismo, pero en lugar de crear siempre el nuevo fichero, si éste ya existiera, se añadiría la información al final del mismo. Con "<" el redireccionamiento se realiza en sentido contrario,

ANOTACIONES

de modo que el contenido del fichero que le indicamos se dirigirá hacia el comando o programa señalado.

Un aspecto muy interesante que debemos conocer es que en sistemas tipo UNIX se separa la salida normal de un programa con la de los errores. Aunque por defecto las dos salidas están dirigidas a la consola donde se ha ejecutado el programa, podemos manipularlas para que se dirijan hacia donde queramos. Para ver esto de manera práctica, probamos de borrar un fichero que no existe con la siguiente instrucción: "`rm fichero > resultados`". Aunque estamos redireccionando la salida del comando hacia el fichero de resultados, por pantalla nos aparecerá un mensaje de error indicando que no se ha encontrado el fichero. Esto es debido a que por defecto los redireccionamientos sólo aceptan la salida estándar del programa y no la de error, que por defecto también se muestra por pantalla. Para redirigir la salida de error, deberíamos indicar, antes del carácter "`>`" el número "`2`", que es la salida de error (la "`1`" es la normal). De esta manera, ejecutando "`rm fichero 2> resultados`" sí que conseguiríamos que la salida se redirigiera al archivo de resultados. También podemos guardar la salida normal y la de errores en dos ficheros diferentes: "`rm fichero 1> resultados 2> errores`". Si por el contrario quisiéramos que todas las salidas se dirigieran hacia un mismo archivo, podríamos utilizar "`>&`". Además, con el carácter "`&`" podemos encaminar salidas de un tipo hacia otras; por ejemplo, si quisiéramos encaminar la salida de errores hacia la normal, podríamos indicarlo del siguiente modo: "`2>&1`".

Es importante tener en cuenta que el orden de los redireccionamiento es significativo: siempre se ejecutan de izquierda a derecha.

2.6.2. Comandos específicos del bash

Aunque algunos de los comandos que hemos visto ya son específicos del bash, este intérprete de comandos dispone de otros que nos pueden servir para realizar otras muchas operaciones interesantes. Un mecanismo muy útil es el de ejecutar procesos en lo que se llama **modo *background***. Este modo indica sencillamente que el proceso se está ejecutando, pero que el *shell* nos devuelve la línea de comandos para poder seguir ejecutando otros programas. Para indicarle esto al bash, debemos escribir el carácter "`&`" después del comando

ANOTACIONES

o programa que vamos a ejecutar. Una vez se ha lanzado el proceso en modo *background,* se muestra una línea donde se nos indica el número de trabajo y PID del proceso lanzado.

Con el comando `jobs` podemos ver qué procesos están lanzados en modo *background* (pasándole el parámetro "`-l`" también podremos ver su PID). Si quisiéramos pasar uno de estos procesos a modo *foreground* (como si lo hubiéramos lanzado desde la línea de comandos sin el carácter "`&`"), podemos utilizar el comando `fg` indicando el PID del proceso. También existe `bg`, que nos envía un determinado proceso a modo *background*. Este último es útil cuando, por ejemplo, ejecutamos un programa en modo *foreground* y lo pausamos con CTRL+Z. Si después ejecutamos `bg` indicándole su PID, el proceso continuará su ejecución en modo *background*. Tal como veíamos en secciones anteriores, los procesos también tienen un jerarquía de padres a hijos. Cuando ejecutamos algún programa en modo *background* no estamos interfiriendo en esta jerarquía, de modo que si salimos de la sesión, todos etos procesos se acabará porque el padre (el intérprete de comandos desde donde los hemos lanzado) ya no estaría en ejecución. Si queremos desvincular un proceso de su padre,podemos utilizar `disown`.

Otro mecanismo muy útil del bash es la historia de comandos. Es normal que utilizando el sistema debamos repetir muchas instrucciones escritas anteriormente. Con las teclas del cursor arriba y abajo podemos ir viendo todos los comandos que hemos ido utilizando y repetir alguno apretando RETURN. También podemos utilizar `history`, con el cual se mostrarán por pantalla todos los comandos ejecutados, enumerados según su aparición. Escribiendo "`!NUM`" se ejecutará el que se corresponda con esta historia. También podemos escribir "`!`" seguido de las letras iniciales de algún programa ejecutado anteriormente y el programa buscará el más reciente para ejecutarlo.

El bash dispone, asimismo, de teclas de acceso rápido que nos permiten ejecutar ciertas acciones sin ni siquiera escribirlas. Algunas de las más frecuentes son:

- TAB: no es necesario escribir el nombre de un fichero, directorio o comando enteramente. Si escribimos los primeros caracteres y después apretamos la tecla del tabulador nos acabará de escribir

Contenido complementario

El bash nos proporciona infinitud de herramientas para regular cualquier aspecto del intérprete de comandos. En su extenso manual podemos encontrar la documentación necesaria para aprender a manipularlas correctamente.

ANOTACIONES

el resto. Si hubiera más de una coincidencia nos mostraría las diferentes posibilidades.

- CTRL+L: limpia la pantalla (igual que el comando `clear`).
- SHIFT+REPÁG: enseña media pantalla anterior.
- SHIFT+AVPÁG: enseña media pantalla posterior.
- CTRL+W: elimina la última palabra escrita.
- CTRL+T: intercambia el orden de los últimos caracteres.
- CTRL+U: borra todos los caracteres anteriores al cursor.
- CTRL+D: sale del intérprete de comandos (equivalente a hacer un `logout`).
- `ulimit` es un comando que nos permite configurar algunos de los aspectos internos relacionados con el bash. Por ejemplo, permite indicar la cantidad de memoria que puede utilizar el intérprete de comandos, el número máximo de archivos que se pueden abrir, etc. Este comando puede servirnos para restringir un poco las acciones que pueden hacer los usuarios de nuestro sistema (en caso de administrar servidores con muchos usuarios).

2.6.3. *Shell scripts* con bash

Los *shell scripts* son ficheros donde escribimos una serie de comandos (cualquiera de los que hemos visto en este capítulo) para que sean ejecutados. Aunque su sintaxis puede llegar a ser muy compleja y tendríamos que entrar en aspectos de programación para entenderla claramente, en esta sección explicaremos de forma resumida algunas de sus características esenciales para que podamos entenderlos y utilizarlos mínimamente (si queremos profundizar más en ellos, podemos recurrir al manual del bash). La primera línea del *shell* script debe especificar el intérprete de comandos que se utiliza:

```
#!/bin/bash
```

Contenido complementario

El comando `fc` nos permite, igual que los shell scripts, escribir una serie de comandos para que se ejecuten pero sin tener que guardar el archivo.

ANOTACIONES

Después de esta línea ya podemos empezar a escribir los comandos que queremos ejecutar, uno en cada línea. Como en todo lenguaje de programación, podemos utilizar variables, estructuras condicionales y bucles. Para declarar una variable utilizaremos la siguiente sintaxis:

```
nombreVariable=contenido
```

Si el contenido es una cadena de caracteres, debemos ponerlo entre comillas, si es un número, no hace falta poner nada y si queremos guardar en la variable la salida de un comando, deberíamos ponerlo entre caracteres. Para referirnos al contenido de la variable en otras instrucciones, siempre debemos preceder al nombre con el carácter "`$`". Para las instrucciones condicionales podemos utilizar las siguientes estructuras:

```
if condicion; then
  instrucciones
else
  instrucciones
fi
```

donde `condición` puede hacer referencia a un archivo, realizar alguna operación de comparación aritmética (entre caracteres "`(( ))`"), etc. De especial utilidad es el comando `test`, que nos permite hacer comprobaciones de archivos, directorios, etc. y nos devuelve un booleano. De este modo, por ejemplo, si quisiéramos realizar una acción u otra según si existiera un determinado archivo, podríamos utilizar la siguiente estructura:

```
if test -f /etc/inittab; then
  echo "El fichero inittab existe."
else
  echo "El fichero inittab NO existe."
fi
```

Otra estructura condicional es la de selección:

```
case palabra in
  caso1)
```

ANOTACIONES

```
     instrucciones
     ;;
  caso2)
     instrucciones
     ;;
*)
     instrucciones
esac
```

En esta estructura se compara `palabra` con `caso1`, `caso2`, etc., hasta encontrar la que coincida, en la que se ejecutarán las instrucciones del caso. Si no se encontrara ninguna, se pasaría a la sección "`*)`", que es opcional. Esta estructura puede irnos muy bien cuando, por ejemplo, queramos que un determinado *script* haga unas acciones u otras según el parámetro que le pasemos. Los parámetros los podemos referenciar a partir de "`$1`" para el primero, "`$2`" para el segundo y consecutivamente. Para los bucles podemos utilizar alguna de las siguientes estructuras:

Contenido complementario

Para escribir comentarios en los *shell scripts* podemos utilizar el carácter "#" seguido del comentario que queramos. Éste será válido hasta final de línea.

```
#BUCLE TIPO FOR
for i in lista; do
  instrucciones
  done
#BUCLE TIPO WHILE
while condición; do
  instrucciones
done
```

Naturalmente, antes de poder ejecutar un *shell script* debemos dar el permiso de ejecución al fichero correspondiente (comando `chmod 750 nombreFichero`).

ANOTACIONES

3. Taller de Knoppix

3.1. Introducción

Este taller pretende llegar a ser vuestra primera experiencia con un entorno UNIX. Por esta razón, su desarrollo es guiado paso a paso, dejando, por supuesto, la puerta abierta a los más curiosos para que investiguen por cuenta propia. Con él se pretende mostrar de forma práctica todo lo que se ha expuesto hasta ahora de forma teórica.

Todo el taller puede desarrollarse sobre cualquier PC, ya que el riesgo de dañar la información que podemos tener es mínimo. Se ha escogido esta distribución porque para arrancarla no se requieren conocimientos previos del sistema operativo, y porque, una vez detenido el sistema, no deja rastro, a no ser que nosotros lo forcemos (KNOPPIX, por defecto, no monta en el sistema los discos duros, así que nuestros datos están a salvo), en el ordenador por donde se ha hecho correr. Obviamente, si se dispone de un sistema operativo tipo UNIX, se puede usar para hacer el seguimiento del taller.

El hecho de ser una distribución arrancable (*bootable*) desde un CD-ROM y no dejar rastro en el ordenador donde se ha ejecutado, una vez ha terminado el proceso de parada, hace que, aun estando basado en Debian, el sistema de ficheros no siga lo que marca la Debian Policy al respecto. No obstante, estas diferencias no afectarán al desarrollo del taller, y, todo lo que aprendamos será válido para los posteriores. Además, es bueno que desde el principio nos acostumbremos a trabajar con distintas distribuciones, y aprendamos a distinguir entre lo que es común a todos los sistemas basados en UNIX y lo que es propio de cada distribución.

Antes de empezar sólo un consejo: adelante con nuestras propias iniciativas, intentemos responder nosotros mismos a nuestras inquietu-

ANOTACIONES

des, consultemos los `man`, hagamos pruebas, fallemos y analicemos el porqué lo hemos hecho, intentémoslo de nuevo, una y otra vez, hasta conseguir los resultados deseados; es así como se aprende UNIX, sin miedo, sacando partido de los propios fallos.

3.2. Arranque del sistema

En primer lugar debemos asegurarnos de que nuestro ordenador arrancará desde el CD-ROM. Para ello, entraremos en la BIOS (*Basic Input Output System*), generalmente pulsando la tecla Supr durante el proceso de chequeo de la memoria RAM, y comprobaremos que el CD-ROM está configurado como primer dispositivo de arranque; si es así, ya podemos salir de la BIOS sin necesidad de guardar nada y poner el CD-ROM de KNOPPIX en el lector. Si no fuese el caso, haríamos los cambios pertinentes y los salvaríamos antes de salir de la BIOS.

Tras reiniciar el ordenador, transcurridos unos segundos nos aparecerá la pantalla de arranque de KNOPPIX con las siguientes líneas en la parte inferior:

```
F2 for help
boot:
```

Podemos pulsar la tecla F2 para entrar en la pantalla donde se nos muestran las opciones que acepta KNOPPIX para arrancar. Podríamos, por ejemplo, arrancar con teclado español y lengua castellana para la interacción, con GNOME como Windows manager y activar el *scroll wheel* del ratón; para hacerlo, bastaría con teclear en la línea de comandos (`boot:`) lo siguiente “knoppix lang=es gnome wheelmouse”. Pero no es el caso; el ejemplo anterior era sólo para mostrar la potencia de KNOPPIX; nosotros, después de insertar en la disquetera un disquete nuevo formateado –sobre el que daremos nuestros primeros pasos en Linux, garantizando así el resto de información que podamos tener en nuestros discos duros– sólo escribiremos “knoppix 2” y pulsaremos INTRO para arrancar el sistema en modo texto.

ANOTACIONES

Inicialmente, el teclado está configurado para EUA (us), así que algunos caracteres no se corresponden con el teclado español; esto lo arreglaremos en seguida; aun así, puede ser de interés saber dónde se encuentran algunos caracteres us en el teclado español: "=" está en la tecla <>, "/" en -_ y "-" en la tecla '?.

Una vez pulsado INTRO, KNOPPIX empezará a cargar el sistema operativo, devolviendo por pantalla algunos de los resultados de los tests que va ejecutando para la autoconfiguración.

Una vez terminado este proceso, se obtendrá la línea de comandos `root@tty1[/]#:`

```
Welcome to the KNOPPIX live Linux-on-CD!

Found SCSI device(s) handled by atp870u.o.
  Accessing KNOPPIX CD-ROM at /dev/scd0...
Total memory found: 515888 kB
Creating /ramdisk (dynamic size=407928k) on /dev/shm...Done.
Creating directories and symlinks on ramdisk...Done.
Starting init process.
INIT: version 2.78-knoppix booting
  Processor 0 is AMD Athlon(TM) XP 2100+ 1732MHz, 256 KB Cache
  APM Bios found, power management functions enabled.
  USB found, managed by hotplug.
  Enabling hotplug manager.
Autoconfiguring devices...                    Done.
  Mouse is Generic 3 Button Mouse (PS/2) at /dev/psaux
  Soundcard: CM8738, driver=cmpci
  AGP bridge detected.
  Video is ATI|Radeon 7500 QW, using XFree86(radeon) Server
  Monitor is LTN 020e, H:31-60kHz, V:55-75Hz
  Using Modes "1024x768" "800x600" "640x480"
Enabling DMA acceleration for: hde.
Scanning for Harddisk partitions and creating /etc/fstab... Done.
  Network device eth0 detected, DHCP broadcasting for IP.
    (Backgrounding)
  Automounter started for: floppy CD-ROM CD-ROM1.
INIT: Entering runlevel: 2
root@tty1[/]#
```

Ya estamos dentro del sistema. En este caso no ha hecho falta ni usuario ni *password*, hemos entrado como *root* directamente; lo no-

ANOTACIONES

tamos porque el *prompt* termina con el carácter "#". Para cualquier otro usuario distinto del de *root*, el último carácter sería "$".

3.3. Paro del sistema

Una vez dentro, lo primero que debemos saber, tal como ya se ha remarcado, es cómo pararlo. Recordemos una vez más que no podemos parar el ordenador sin haber detenido antes el sistema operativo. Hay múltiples maneras de hacerlo. Las más comunes son: pulsar la combinación de teclas CTRL+ALT+Supr o mediante el comando `halt` (hay muchas más, mediante el comando `reboot`, cambiando de *runlevel* a 0 o 6, etc.). Una vez hayamos dado la orden al sistema para que se detenga, éste empezará a ejecutar las instrucciones pertinentes de paro (desmontado de dispositivos, parada de procesos, etc.), y al final de todo este proceso, KNOPPIX expulsará el CD-ROM , y nos pedirá que pulsemos INTRO para parar el ordenador.

3.4. Configuración del teclado

Una vez sabemos cómo parar el sistema y volvemos a estar en él siguiendo los mismos pasos de arranque anteriores, lo primero que debemos hacer es configurar el mapeo del teclado correctamente. Hay dos formas de hacerlo, manualmente:

```
root@tty1[/]#loadkeys /usr/share/keymaps/i386/qwerty/es.kmap.gz
```

o bien gráficamente con el comando `kbdconfig` y escogiendo la opción "`es      es`".

Es importante recordar que siempre tenemos la opción de autocompletar pulsando TAB una vez, si sólo hay una opción posible, y que se autocomplete; o dos veces seguidas, si hay más de una opción, y que se nos muestren las posibilidades. Podemos probarlo con el comando loadkeys mismo. Tecleamos sólo una "`l`" y pulsamos TAB

ANOTACIONES

una vez, el sistema emite un pitido; y una segunda para obtener lo siguiente:

```
root@tty1[/]# l
Display all 143 possibilities? (y or n)
```

En este caso no queremos que se nos muestren las 143 posibilidades. Pulsamos, pues, "n", añadimos una "o" y repetimos la operación anterior de pulsar dos veces TAB. Ahora obtenemos un resultado distinto:

```
root@tty1[/]# lo
loadkeys        locale-gen   lockfile-create  logger   logredo     lookbib
loadshlib       localedef    lockfile-remove  login    logresolve  lorder
local           locate       lockfile-touch   logname  logrotate   losetup
locale          lockfile     logdump          logout   look
root@tty1[/]# lo
```

Vemos que sólo falta añadir "adk" y volver a pulsar TAB para que funcione el autocompletado obteniendo:

```
root@tty1[/]# loadkeys
```

El autocompletado también sirve para hacer referencia a ficheros y directorios. Basta con teclear "loadkeys /u" y pulsar TAB para obtener:

```
root@tty1[/]# loadkeys /usr
```

y volver a pulsar TAB para obtener:

```
root@tty1[/]# loadkeys /usr/
```

El autocompletado es una herramienta muy útil, no sólo porque ahorra teclear, sino porque además sirve para verificar que hemos escrito correctamente tanto los comandos, como los directorios y ficheros. Es una herramienta que, una vez acostumbrados a usarla, su ausencia se hace notar.

ANOTACIONES

3.5. Inspección del sistema

Una vez tenemos el teclado configurado, podemos proceder a inspeccionar un poco el sistema. En primer lugar, ¿en qué sitio de la estructura de directorios nos encontramos actualmente? Lo podemos saber mediante el comando `pwd`:

```
root@tty1[/]# pwd
/
```

Estamos en la raíz del sistema, lo sabemos por el retorno del comando `pwd`, pero también lo hubiéramos podido saber leyendo la información que nos da el*prompt*: "`root@tty1[/]#`". Anteriormente ya hemos entendido el significado del carácter "`#`", conozcamos ahora el resto de información que éste nos da:

"`root`", en este campo aparece el nombre de usuario, *root* en este caso, información un tanto redundante, pues de la misma manera indica el carácter "#", pero para el resto de usuarios es interesante.

Después del carácter "`@`", que simplemente sirve para separar campos, hay "`tty1`", este campo nos indica en qué terminal nos encontramos, y es que, tal como ya se ha dicho, Linux es multiusuario y multiproceso; así pues, no es de extrañar que podamos acceder al sistema desde varios terminales. Concretamente KNOPPIX ofrece, por defecto, cuatro terminales, a los que podemos acceder mediante las combinaciones de teclas ALT+F1 (tty1), ALT+F2 (tty2), ALT+F3 (tty3) y ALT+F4 (tty4); esto es extremadamente útil, ya que nos permite tener hasta cuatro sesiones de trabajo, pudiendo, por ejemplo, estar probando un comando en una, en otra tener el man del comando en cuestión para ir leyendo sus opciones, mientras que en otra podemos tener un editor abierto para ir tomando nota de lo que vamos haciendo (muchas distribuciones ofrecen seis terminales modo texto, reservando el séptimo para el terminal gráfico; en KNOPPIX el terminal gráfico, si se ha habilitado, y no es nuestro caso, se hallaría en el quinto terminal). Así pues, probamos la combinación ALT+F2, observamos ahora que en el campo que estamos estudiando aparece "`tty2`", y ejecutamos un comando, por ejemplo man man, para leer el man de la aplicación man.

ANOTACIONES

De momento la dejamos ahí y volvemos a la tty1, y comprobamos que todo está como lo habíamos dejado.

Finalmente, y siguiendo con la interpretación del *prompt*, antes del campo ya conocido "#"encontramos "`[/]`": aquí se nos indica el directorio actual, el raíz en este caso, tal y como nos había informado el comando `pwd`. La diferencia radica en que el *prompt* sólo nos informa del directorio actual, mientras que `pwd` nos devuelve la ruta completa.

Podemos ahora listar los contenidos del directorio en el que estamos. Para hacerlo, utilizaremos el comando ls, primero sin pasarle ninguna opción, y después dos para indicarle que nos devuelva información más detallada (normalmente las opciones van precedidas de "–", aunque hay comandos que no lo exigen, y una vez puesto el "–" se pueden concatenar como se hace en el ejemplo):

```
root@tty1[/]# ls
KNOPPIX bin boot CD-ROM dev etc home lib mnt opt proc
ramdisk sbin tmp usr var
root@tty1[/]# ls -la
total 45
drwxr-xr-x    9     root    root    1024 Mar 2 18:37 .
drwxr-xr-x    9     root    root    1024 Mar 2 18:37 ..
drwxr-xr-x    20    root    root    4096 Jan 18 18:05 KNOPPIX
lrwxrwxrwx    1     root    root    12 Jan 18 17:58 bin -> /KNOPPIX/bin
drwxr-xr-x    9     root    root    1024 Mar 2 18:37 .
lrwxrwxrwx    1     root    root    13 Jan 18 17:58 boot -> /KNOPPIX/boot
dr-xr-xr-x    5     root    root    2048 Jan 4 03:34 CD-ROM
drwxr-xr-x    3     root    root    29696 Mar 2 17:58 dev
drwxr-xr-x    139   root    root    6144 Mar 2 17:49 etc
lrwxrwxrwx    1     root    root    13 Mar 2 18:37 home -> /ramdisk/home
lrwxrwxrwx    1     root    root    12 Jan 18 17:58 lib -> /KNOPPIX/lib
drwxr-xr-x    13    root    root    1024 Mar 2 17:38 mnt
lrwxrwxrwx    1     root    root    12 Jan 18 17:58 opt -> /KNOPPIX/opt
dr-xr-xr-x    30    root    root    0 Mar 2 18:37 proc
drwxrwxrwt    4     root    root    80 Mar 2 18:37 ramdisk
lrwxrwxrwx    1     root    root    13 Jan 18 17:58 sbin ->/KNOPPIX/sbin
lrwxrwxrwx    1     root    root    8 Jan 18 17:58 tmp-> /var/tmp
lrwxrwxrwx    1     root    root    12 Jan 18 17:58 usr -> /KNOPPIX/usr
```

Fijémonos en los distintos colores con los que se nos muestran los resultados (podemos observar los colores citados en nuestro PC a medida que vayamos progresando en el taller de Knoppix): azul marino para los directorios; magenta para los enlaces simbólicos, cuyo destino se nos muestra después de la combinación de caracteres "->"; verde para los *scripts* y ejecutables (aunque, obviamente, en el directorio raíz no esperamos encontrar ninguno de ellos, ya que, tal como hemos visto, en UNIX el orden dentro del sistema de ficheros es muy rígido), etc.

Entremos en algún directorio donde seguro que encontraremos ejecutables, por ejemplo `/usr/bin/` (para hacerlo, ejecutemos "`cd /usr/bin/`"). El directorio `/usr/` es en realidad un enlace simbólico sobre `/KNOPPIX/usr/`; podríamos, pues, acceder a sus contenidos entrando en él directamente mediante el comando `cd`, o bien seguir mediante el mismo comando el destino real. Sin embargo, escogemos la primera opción, pues cuando nos encontremos en un sistema UNIX instalado en un disco duro este directorio será un directorio real:

```
root@tty1[bin]# ls -la
total 171831
drwxr-xr-x      2 root     root     311296 Jan 18 18:04 .
drwxr-xr-x      14 root    root     2048 Aug 17 2002 ..
-rwxr-xr-x      1 root     root     5380 Apr 1 2002 411toppm
-rwxr-xr-x      1 root     root     1649 Sep 15 20:36 822-date
-rwxr-xr-x      1 root     root     1971 Jun 5 2002 AbiWord
.
.
.
-rwxr-xr-x      1 root     root     17992 Sep 16 22:34 zsoelim
-rwxr-xr-x      1 root     root     966 Sep 4 13:53 zxpdf
```

ANOTACIONES

Probablemente no hayamos podido ver más que las últimas líneas de las que han cruzado la pantalla. Podemos visualizar algunos resultados más pulsando Shift+RePág para retroceder en listado y Shift+AvPág para avanzar; aun así, el *búffer* del terminal tiene un límite, y probablemente tampoco podremos visualizar, mediante esta técnica, toda la información devuelta por el comando `ls`. Debemos, pues, recurrir a otras técnicas, el redireccionamiento de la salida (en

este caso sobre un fichero) o el uso de *pipes* "`|`" y paginadores (`less` en este caso, que usa las teclas RePág y AvPág para desplazar el contenido mostrado por pantalla, y la tecla "`q`" para salir). Utilizaremos esta última, ya que no nos interesa para nada guardar un listado del contenido de un directorio, aprovecharemos, asimismo, para poner en práctica otra utilidad que nos ahorrará teclear: pulsando los cursores de arriba y abajo podemos desplazarnos por todas las líneas de comandos que hemos pasado al sistema; así pues, para obtener la línea deseada, "`ls -la | less`", bastará con pulsar una vez el cursor de arriba y añadir "`| less`".

`less` es un paginador muy potente que KNOPPIX, y la mayoría de distribuciones, usa para mostrar los contenidos de los man (aunque, como la mayoría de servicios en Linux, lo podemos configurar a nuestro gusto). Así pues, podemos hacer un "`man less`" para informarnos un poco sobre este comando, y de paso irnos acostumbrando a la morfología de esta ayuda. Obviamente, ya que estamos bajo `less`, para salir de man basta con pulsar la tecla "`q`".

Fijémonos en que en todos los listados de contenidos de directorios que hemos ido haciendo siempre aparecen, al principio de todo, dos directorios un tanto especiales: "`.`" y "`..`"; al crear directorios, éstos son creados automáticamente por el sistema como subdirectorios del directorio creado, y contienen información muy especial: "`.`" hace referencia al propio directorio (de modo que, si hacemos, por ejemplo, un "`ls -la .`",obtendremos la misma información que haciendo un "`ls -la`") y "`..`" hace referencia al directorio padre (haciendo un "`ls -la ..`" obtendremos el listado de contenidos del directorio padre).

Actividad

5. Hacer un `man` del comando `ls` y entender qué funciones tienen los parámetros "`-a`" y "`-l`".

¿Recordamos dónde estamos dentro del sistema de ficheros? ¿No? Pues el comando `pwd` nos lo recordará. Volvamos al directorio raíz. Hay básicamente dos formas de hacerlo, y otra especial para este caso; de cualquier manera, todas se basan en el uso del comando `cd`, diferenciándose entre ellas por el argumento que le pasamos en cada uno de los casos. El primer modo de ir es volviendo a los direc-

ANOTACIONES

torios padre paso a paso, mediante “`cd ..`” (atención, el sistema no entiende “`cd..`”, ya que interpreta que estamos intentando ejecutar un comando llamado “`cd..`”, que no existe), iterativamente hasta llegar al directorio raíz (lo veremos en el *prompt*). La segunda es más eficaz, ya que con una sola línea de comandos conseguiremos nuestro propósito, ir al directorio deseado, pasándolo como argumento al comando cd, “`cd /`” en este caso; esta segunda opción es mucho más potente, ya que permite pasar directamente a directorios que pertenecen a distintas ramas, podríamos, por ejemplo, haber hecho “`cd /etc/rcS.d/`” (la última barra es opcional) para ir a este directorio, estando en `/usr/bin/`. Como hemos dicho, para este caso hay una tercera opción para conseguir el mismo fin y con variaciones: se trata de ejecutar `cd` sin pasarle ningún argumento, y nos situaríamos directamente en la raíz del sistema, ya que éste es, en el caso de KNOPPIX (caso excepcional y un tanto peculiar, ya que en la mayoría de distribuciones, y según el Filesystem Hierarchy Standard (http://www.pathname.com/fhs), *root* debe tener un *home* propio `/root`), el directorio *home* de *root*; una variación de este método es “`cd`   ”, ya que “ ” es equivalente al *home* del usuario, el directorio raíz en este caso. Aún existe otra opción para volver al directorio raíz, o mejor dicho, para volver al directorio del que venimos, el raíz, ya que habíamos ejecutado “`cd /usr/bin`”, y que es “`cd –`”, ya que en “–” se guarda el último directorio donde hemos estado antes del actual.

3.6. Manejo de directorios y ficheros

Una vez examinadas diferentes opciones para hacer lo mismo, nos hallamos todavía en el directorio raíz. Podemos aprender ahora a crear directorios y ficheros, moverlos, copiarlos, borrarlos, etc. Lo haremos sobre el disquete que hemos introducido al arrancar KNOPPIX, y que el sistema mismo, al arrancar, lo ha automontado (más adelante aprenderemos cómo montar y desmontar dispositivos). Recordemos que en UNIX los dispositivos antes de poderse usar deben ser montados, y que antes de retirar el soporte deben ser desmontados. Este último punto es extremadamente importante, ya que si no, la integridad de los datos no está en absoluto garantizada. Podemos, por ejemplo, antes de proceder a trabajar sobre el disquete, intentar retirar el CD-ROM pulsando el botón de

ANOTACIONES

expulsión. ¡Oh, sorpresa!, no se puede; no pensemos que se ha estropeado el dispositivo y menos que lo haya hecho Linux. Simplemente sucede que este dispositivo también ha sido montado automáticamente por el sistema durante el arranque, y que, por tanto, ha pasado a formar parte de éste; no tendría ningún sentido que pudiésemos retirar el CD-ROM sin antes informar al sistema; así pues, éste ha tomado el control de este dispositivo y, entre otras cosas, ha deshabilitado el botón de expulsión del CD-ROM, precisamente con el fin de que accidentalmente éste sea retirado sin antes ser desmontado. No sucede lo mismo con la disquetera: en este dispositivo la expulsión se hace de forma puramente mecánica, así que es imposible que el sistema pueda impedir que nosotros expulsemos el disquete sin informarle antes, pero esto no es en absoluto recomendable, tal como ya se ha dicho, porque esto puede suponer la pérdida de todos los datos que pudiese contener. Así pues, mientras no aprendamos a montar y desmontar dispositivos de soporte de datos, el disquete debe permanecer en la disquetera desde que el sistema arranca hasta que está parado, ya que durante el proceso de parada, entre otras operaciones, se efectuarán las de desmontado de estos dispositivos. Vayamos al disquete: pero, ¿dónde está?, ¿dónde lo ha montado el sistema? Para responder a estas preguntas ejecutamos el comando `mount` sin argumentos ni opciones adicionales (precisamente éste es el comando que se utiliza para montar dispositivos, y `umount` para desmontarlos):

```
root@tty1[/]# mount
/dev/root on / type ext2 (rw)
/dev/CD-ROM on /CD-ROM type iso9660 (ro)
/dev/cloop on /KNOPPIX type iso9660 (ro)
/dev/shm on /ramdisk type tmpfs (rw,size=407928k)
none on /proc/bus/usb type usbdevfs (rw,devmode=0666)
automount(pid443) on /mnt/auto type autofs (rw,fd=6,pgrp=443,minproto=2,maxproto=4)
/dev/fd0 on /mnt/auto/floppy type vfat (rw,nosuid,nodev,uid=1000,gid=1000,umask=000)
```

Este comando, llamado sin argumentos, nos muestra los dispositivos montados en el sistema en el momento de ejecutarlo junto con alguna información adicional acerca de ellos (esta misma información se puede encontrar en el fichero `/etc/mtab`; en consecuencia, haciendo un cat "`/etc/mtab`" conseguiríamos los mismos resultados). En la segunda línea retornada por el comando ya podemos ver que, efectivamente, el CD-ROM, `/dev/CD-ROM`, está montado en el sistema, y además podemos ver que está

ANOTACIONES

montado sobre el directorio /CD-ROM/. La última línea retornada responde a las preguntas que nos formulábamos anteriormente: el sistema ha montado el disquete, /dev/fd0/, en /mnt/auto/floppy/. Cambiamos pues a este directorio para empezar a trabajar sobre el disquete y comprobamos que efectivamente está vacío:

```
root@tty1[/]# cd /mnt/auto/floppy
root@tty1[floppy]# ls -la
total 8
drwxrwxrwx 3 knoppix knoppix 7168 Jan 1 1970 .
drwxr-xr-x 3 root    root    0 Mar 3 19:34 ..
```

Creamos nuestro primer directorio, entramos en él y creamos un par de subdirectorios:

```
root@tty1[floppy]# mkdir dir00
root@tty1[floppy]# cd dir00/
root@tty1[dir00]# mkdir subdir00 subdir01
root@tty1[dir00]# ls
subdir00 subdir01
```

Entramos en el primer subdirectorio y creamos nuestro primer fichero:

```
root@tty1[dir00]# cd subdir00
root@tty1[subdir00]# touch file00
root@tty1[subdir00]# ls -la
total 1
drwxrwxrwx 2 knoppix knoppix 512 Mar 3 20:21 .
drwxrwxrwx 4 knoppix knoppix 512 Mar 3 20:21 ..
-rwxrwxrwx 1 knoppix knoppix   0 Mar 3 20:21 file00
```

En realidad *touch* no sirve para crear ficheros vacíos, aunque lo hace si el fichero no existe, sino que sirve para cambiar las informaciones relativas a fechas y horas de los archivos. Podemos poner algún contenido a nuestro primer fichero y comprobar que efectivamente ha quedado registrado:

```
root@tty1[subdir00]# echo “my fist hello world in Linux” > file00
root@tty1[subdir00]# cat file00
my first hello world in Linux
```

ANOTACIONES

Le añadimos un poco más de texto:

```
root@tty1[subdir00]# echo "Nice to meet you, we're gonna be good friends" >> file00
root@tty1[subdir00]# cat file00
my first hello world in Linux
Nice to meet you, we're gonna be good friends
```

No olvidemos usar los cursores para ahorrarnos tecleo, ni del autocompletado. Ahora podemos utilizar un editor de texto para crear nuestro segundo fichero. Para ello usaremos el `vi`. Recordemos que este editor tiene dos modos: el de comandos, en el que se entra cuando se arranca, y el modo de edición. Para entrar en el modo de edición basta con pulsar la tecla "`i`", y para pasar al modo de comandos hay que pulsar ESC. Una vez en modo comando "`w`" para guardar y "`q`", para salir (obviamente, `vi` dispone de muchísimos más comandos, pero por ahora con estos dos nos basta). Es muy interesante conocer estos comandos básicos de `vi`, ya que este editor viene en casi todos los paquetes básicos de instalación de cualquier distribución, y si ésta fallase en algún momento, nos puede ser útil para modificar algún fichero y proseguir dicha instalación. Ejecutamos, pues, `vi` seguido del nombre que queremos dar a nuestro segundo fichero, pulsamos la tecla "`i`", escribimos lo que nos parezca oportuno, pulsamos ESC y "`:wq`" para guardar y salir. Mediante `more`, otro paginador, comprobamos que todo ha ido como esperábamos:

Nota

En sistemas básicos tipo UNIX, si no se cuenta con `vi` se puede probar si se encuentran instalados otros editores de texto más elementales como pueden ser `nano`, `pico`, etc.

```
root@tty1[subdir00]# vi file01
it seems we're on the right way, mate!!!
I agree.
~
~
~
:wq
root@tty1[subdir00]# more file01
it seems we're on the right way, mate!!!
I agree.
```

Ahora intentamos borrar nuestro primer fichero. Lo haremos mediante el comando `rm`:

```
root@tty1[subdir00]# rm file00
rm: remove regular file 'file00'?
```

ANOTACIONES

Lo cierto es que esta pregunta por parte del sistema no nos la esperábamos. Pulsamos CTRL-C para cancelar, "n" para no borrar o "y" para hacerlo. Lo importante ahora es que entendamos por qué el sistema nos ha formulado esta pregunta. Si leemos el man del comando `rm` veremos que éste debería borrar el archivo sin hacer ninguna pregunta (recordemos que podemos ejecutar dicho `man` en otra tty, y que en la tty2 aún tenemos abierto el man de man, si no hemos salido de él mediante "q", o si no hemos apagado el ordenador desde que lo invocamos, y proseguir así nuestro trabajo en la tty en que estábamos). Pero fijémonos en la explicación de la opción "-i" de este mismo `man`. ¿Qué está pasando? Parece como si esta opción estuviera activada por defecto. En realidad no es exactamente así, lo que ocurre es que el sistema por defecto ha establecido unos alias; para verlos todos, ejecutemos el comando `alias`:

```
root@tty1[etc]# alias
alias ..='cd ..'
alias cp='cp -i'
alias l='ls -a --color=auto'
alias la='ls -la --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mv='mv -i'
alias rm='rm -i'
alias where='type -all'
alias which='type -path'
```

Aquí empezamos a entender muchas más cosas, como por ejemplo por qué el retorno del comando `ls` es en colores, o que para hacer un "`ls -la`" basta con teclear `la`. Y también entendemos por qué, cuando hemos ejectuado el `rm` anterior, el sistema nos ha preguntado si realmente queríamos hacerlo. Mediante `alias` podemos establecer y modificar el comportamiento por defecto de comandos o incluso crear otros nuevos:

```
root@tty1[subdir00]# alias hi='echo¨I say hello"'
root@tty1[subdir00]# hi
I say hello
```

Continuando con el `man` del comando `rm`, nos disponemos a leer para qué sirven las opciones `-f` y `-r`. La primera sirve para que el comando se ejecute sin mostrar *prompt*, o lo que es lo mismo, des-

ANOTACIONES

obedeciendo la opción `-i`, así que, si no habíamos borrado nuestro primer fichero, podemos hacerlo ahora mediante “`rm -f file00`”. La segunda opción fuerza la recursividad, es decir, que la orden se extienda sobre los posibles subdirectorios y sus contenidos. Estas opciones son comunes a la mayoría de comandos básicos destinados a la manipulación de directorios y ficheros; así pues, podemos crear un segundo directorio en la raíz del disquete con todos los contenidos del primero que hemos creado mediante el comando `cp`; para hacerlo, debemos acudir a la recursividad:

```
root@tty1[subdir00]# cp /mnt/auto/floppy/dir00/ /mnt/auto/floppy/dir01 -r
```

En este caso hemos usado el direccionamiento absoluto –tanto para especificar el origen como el destino– para realizar la operación; es decir, hemos indicado, partiendo del directorio raíz, la ruta completa, tanto para el origen como para el destino. Del mismo modo, podríamos haber utilizado el direccionamiento relativo para especificar el origen de la operación, su destino o ambas cosas. Cuando usamos el direccionamiento relativo, el origen del direccionamiento es la posición actual dentro del filesystem. Como en la mayoría de las ocasiones, el sistema nos ofrece la posibilidad de obtener los mismos resultados empleando distintos métodos. Se trata de conocer cuantos más mejor, y saber escoger el más efectivo para cada caso en particular. Así pues, hubiésemos obtenido el mismo resultado haciendo “`cp ../../dir00/ ../../dir01 -r`”, es decir, comunicándole a `cp` que el origen es el directorio `/dir00/`, el cual se encuentra dos ramas por debajo de nuestra posición actual, y que el destino es `/dir01/`, al que también queremos situar dos ramas por debajo de nuestra posición. Asimismo, hubiese sido perfectamente válida la línea “`cp -r ../ ../../dir02/`” para alcanzar los mismos fines.

Actividad

6. Explicar el porqué de esta última aseveración.

Ahora podemos descender un subdirectorio, con lo cual nos situaremos en `/mnt/auto/floppy/dir00/` y podemos copiar el segundo fichero que hemos creado en el subdirectorio `/subdir00` en este mismo directorio:

```
root@tty1[dir00]# cp subdir00/file01.
```

ANOTACIONES

Debemos entender completamente el significado y el porqué de la línea anterior: en primer lugar especificamos el origen del fichero que queremos copiar (también hubiese sido válido, entre muchas otras opciones, "`./subdir00/file00`"), y en segundo lugar hay que especificar obligatoriamente el destino, siendo éste la posición actual, es decir "`.`". Mediante un ls podemos comprobar si hemos obtenido el resultado deseado. Ahora podemos situarnos en el directorio padre, donde se halla montado el disquete, y borrar todo lo que hemos generado hasta ahora. Para hacerlo, utilizaremos el *wildcard* "`*`" (existe también el *wildcard* ?, el cual sirve para un solo carácter, así como distintos métodos para especificar rangos de caracteres, y para referirse, en general, a más de un fichero o directorio) para ahorrar tecleo:

```
root@tty1[floppy]# rm -rf *
```

Nota

Mucho cuidado al ejecutar el comando rm-rf* como root ya que puede tener consecuencias nefastas.

Actividad

7. Leer el man de rm y entender porqué "rm-rf*" es tan peligroso.

3.7. Administración de usuarios

Debe alarmarnos el hecho de haber estado realizando todas las tareas anteriores como *root*, puesto que ya tenemos muy claro que este usuario sólo debe emplearse en caso necesario.

No obstante, queda justificado, ya que hasta ahora no habíamos tenido ningún contacto directo con un sistema UNIX, y esto nos ha servido para familiarizarnos un poco con los comandos básicos y con su sistema de ficheros. Ha llegado el momento de empezar a trabajar como hay que hacerlo siempre a partir de ahora, es decir, utilizando la cuenta de *root* sólo cuando es estrictamente necesario, como en el primer paso que daremos a continuación, crear un nuevo usuario del sistema y asignarle una cuenta, ya que esta operación sólo la puede realizar el *root*. Procedemos pues a crear un usuario.

```
root@tty1[etc]# useradd user00
```

Hemos creado nuestro primer usuario, pero el proceso ha sido un tanto opaco pues el sistema no nos ha retornado ninguna informa-

ción respecto a este comando que acabamos de ejecutar. No sabemos pues ni a qué grupo o grupos pertenece este usuario, dónde tiene éste el *home*, qué *shell* se le ha asignado por defecto, etc. Mucha de esta información la encontraremos en el fichero de passwords, `/etc/passwd`, así que analicemos, por campos, su contenido:

```
root@tty1[/]#cd /etc
root@tty1[etc]# cat passwd
root:x:0:0:root:/home/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
.
.
.
partimag:x:104:65534::/home/partimag:/bin/false
user00:x:1001:100::/home/user00:/bin/bash
```

La información deseada la encontramos en la última línea del fichero, pues nuestro usuario ha sido el último en añadirse. Analicemos el contenido de esta línea:

- `user00` es el nombre del usuario, que ya conocíamos, ya que lo hemos creado nosotros mismos.
- `x` su *password* se halla en el fichero de shadow, `/etc/shadow`.
- `1001` es su número de identificación como usuario (UID).
- No tiene ningún comentario asociado.
- `/home/user00` éste es su directorio *home*, tal como era de esperar, pues todos los usuarios tienen su *home* en `/home/usuario`.
- `/bin/bash` el *shell* con el cual el usuario se encontrará al iniciar la sesión es el bash, el mismo que hemos utilizado hasta ahora.

La pregunta que nos formulamos inmediatamente ante estas informaciones es relativa al *password* del usuario. Debemos pues

ANOTACIONES

consultar el fichero de shadow, ya que así lo indica el fichero /etc/passwd. De paso, introduciremos un concepto nuevo, el de filtro, que en este caso se materializa con el comando grep (*general regular expression processor*). Los filtros son herramientas extremadamente potentes y muy usadas en UNIX, y su campo de acción se extiende mucho más allá de lo que comúnmente se entiende por filtro. En este caso, pasaremos, mediante un *pipe*, la salida de cat a grep, para que éste muestre por pantalla aquellas líneas que contengan la palabra que le pasamos como parámetro, user00:

```
root@tty1[etc]# cat shadow | grep user00
user00:!:12115:0:99999:7:::
```

A partir de ahora no debemos preocuparnos más: el segundo campo nos indica que nuestro usuario no tiene *password*. Si lo hubiese tenido, no lo podríamos conocer, pues tal como se ha dicho, los *passwords* se guardan encriptados y la encriptación es unidireccional. Pero esto tampoco debería preocuparnos, ya que *root*, mediante el comando passwd, puede cambiar el *password* de cualquier otro usuario. Así pues, lo único que realmente debemos intentar es no olvidar el *password* de *root,* que tampoco tiene *password* en este caso, tal como shadow nos indica, aunque si esto sucediera, también tiene solución. Continuemos con nuestra investigación acerca de las propiedades de nuestro nuevo usuario. /etc/group nos ayudará a saber exactamente a qué grupos pertenece:

```
root@tty1[etc]# cat group | grep user00
root@tty1[etc]# cat group | grep 100
users:x:100:knoppix
knoppix:x:1000:
```

Con la primera línea de comandos nos preguntamos, con respuesta negativa, si /etc/group tiene alguna referencia explícita a user00 (también se podría haber hecho mediante el comando groups, “groups user00”). Con la segunda buscamos el reverso del grupo 100, grupo primario de user00 según /etc/passwd, grupo que se llama users en este caso. Así pues, ya podemos afirmar que nuestro nuevo usuario pertenece a un solo grupo, y que éste es user. El fichero /etc/group también se puede editar para añadir usuarios

ANOTACIONES

a grupos y crear grupos nuevos, pero hay que remarcar que en ningún caso en el campo de usuarios pertenecientes a un grupo en concreto se puede poner el nombre de otro grupo con la intención de añadir todos los usuarios de este último al primero. Pero esta práctica no es demasiado recomendable, ya que existen comandos específicos para trabajar con grupos (`newgrp`, `addgroup`, etc.)

Queda sólo una última cuestión a resolver: si listamos los contenidos de `/home/`, veremos que el subdirectorio `/user00/` no existe. Debemos, pues, crearlo y asignarle las propiedades y atributos pertinentes manualmente:

```
root@tty1[etc]#mkdir /home/user00
root@tty1[etc]#chown user00 -R /home/user00
root@tty1[etc]#chgrp users -R /home/user00
root@tty1[etc]# cd /home
root@tty1[home]# ls -la
total 0
drwxr-xr-x  5 root    root    100  Mar  4  08:12  .
drwxrwxrwt  4 root    root     80  Mar  4  08:35  ..
drwxr-xr-x  2 knoppix knoppix  40  Mar  4  08:35  knoppix
drwxr-xr-x  2 root    root     40  Mar  4  08:35  root
drwxr-xr-x  2 user00  users    60  Mar  4  08:12  user00
```

Nota

El proceso de adición de nuevos usuarios al sistema mediante Debian es mucho más sencillo, como se verá más adelante. Pero crear uno mediante KNOPPIX no ha sido una tarea en balde, pues nos ha servido para familiarizarnos con el sistema y aprender comandos nuevos.

Actividad

8. Mediante man, comprender el funcionamiento de los comandos `chown` y `chgrp`.

Ha llegado el momento de entrar en el sistema como nuevo usuario. Lo haremos mediante el comando `su`, el cual abre un proceso hijo con un login con el usuario que le hayamos pasado. Como root podemos utilizar siempre este mecanismo para entrar como otro usuario sin necesidad de conocer su *password,* ya que al ejecutar `su` como root éste no es necesario; además, se da el caso que el usuario que hemos creado no tiene *password*, y por tanto éste nunca se pe-

ANOTACIONES

dirá. Pero, si probásemos de ejecutar `su` en cualquier otra circunstancia sí que se nos pediría el *password* del usuario:

```
root@tty1[home]# su user00
su(pam_unix)[4312]: session opened for user user00 by (uid=0)
user00@tty1[home]$
```

Notemos primeramente el cambio de aspecto del *prompt*. Hemos dejado de ser usuario root, para pasar a ser user00, por lo que hemos perdido los privilegios de root, hecho que se manifiesta por el cambio del último carácter del *prompt*. Podemos acceder ahora a entrar en nuestro directorio home, creemos un fichero y cambiémosle los atributos para que tan sólo user00 pueda leerlo, ejecutarlo y escribir en él. De momento, el permiso de ejecución no tiene mucho sentido para nosotros, pues no sabemos aún crear ficheros ejecutables, pero es bueno conocerlo de antemano para cuando se precise de él:

```
user00@tty1[home]$ cd
user00@tty1[user00]$ echo "only user00 can read, write and execute this file." > user00file
user00@tty1[user00]$ chmod 700 user00file
```

Actividad

9. Crear un nuevo usuario siguiendo los pasos anteriormente descritos, y comprobar que, efectivamente, este nuevo usuario no puede ni leer el fichero recién creado ni escribir en él.

3.8. Gestión de procesos

UNIX se caracteriza por una excelente gestión de los procesos que se ejecutan sobre el sistema. En primer lugar debemos aprender a detectar qué procesos están corriendo sobre el sistema y sus particularidades (modo en que corren, recursos que consumen, quién los está ejecutando, etc.). Ejecutaremos el comando `ps` (*process status*) pasándole diversos parámetros para ver cómo inciden estos sobre la información devuelta por el comando:

```
root@tty1[/]# ps
PID TTY TIME CMD
481 tty1  00:00:00 bash
        1559 tty1      00:00:00 ps
```

ANOTACIONES

Sin argumentos `ps` nos informa sobre los procesos que corren sobre el terminal donde se ejecuta; naturalmente el primer proceso siempre corresponderá al *shell*.

Podríamos mandar ejecutar un proceso en *background*, `sleep`, por ejemplo (proceso que simplemente espera que transcurra el número de segundos que le pasemos como parámetro para terminar) y observar el retorno de `ps`:

```
root@tty1[/]# sleep 300 &
[1] 1703
root@tty1[/]# ps
  PID TTY      TIME CMD
  481 tty1 00:00:00 bash
 1703 tty1 00:00:00 sleep
 1705 tty1 00:00:00 ps
```

Podemos ahora preguntarnos por todos los procesos que están corriendo sobre el sistema:

```
root@tty1[/]# ps aux
USER  PID  %CPU %MEM  VSZ  RSS  TTY  STAT  START TIME  COMMAND
root  1    0.4  0.0   72   72   ?    S     15:41 0:07  init [2]
root  2    0.0  0.0   0    0    ?    SW    15:41 0:00  [keventd]
root  3    0.0  0.0   0    0    ?    SWN   15:41 0:00  [ksoftirqd_CPU0]
root  4    0.0  0.0   0    0    ?    SW    15:41 0:00  [kswapd]
root  5    0.0  0.0   0    0    ?    SW    15:41 0:00  [bdflush]
root  6    0.0  0.0   0    0    ?    SW    15:41 0:00  [kupdated]
root  52   0.0  0.0   0    0    ?    SW    15:41 0:00  [kapmd]
root  59   0.0  0.0   0    0    ?    SW    15:41 0:00  [khubd]
root  433  0.0  0.1   1368 616  ?    S     15:41 0:00  pump -i eth0
root  475  0.0  0.1   1316 596  ?    S     15:41 0:00 /usr/sbin/automount
root  481  0.0  0.3   2864 1988 tty1 S     15:41 0:00  /bin/bash -login
root  482  0.0  0.3   2864 1988 tty2 S     15:41 0:00  /bin/bash -login
root  483  0.0  0.3   2856 1952 tty3 S     15:41 0:00  /bin/bash -login
root  484  0.0  0.3   2856 1976 tty4 S     15:41 0:00  /bin/bash -login
root  2086 0.0  0.3   3436 1552 tty1 R     16:06 0:00  ps aux
```

También puede ser interesante en determinadas situaciones ejecutar el comando top, el cual nos muestra la actividad de la CPU, el uso de memoria, etc. de forma interactiva.

ANOTACIONES

Ahora vamos a arrancar un proceso, detenerlo, y mandarlo a ejecutar en *background*. Para hacerlo, haremos un “`sleep 20`” y pulsaremos la combinación de teclas CTRL+Z antes de 20 segundos, para pararlo, jobs para asegurarnos de que efectivamente está parado y conocer su número de identificación, y `bg` junto a su número de identificación para mandarlo ejecutar en modo *background*:

```
root@tty1[/]# sleep 20
<Ctrl+Z>
[1]+ Stopped            sleep 20
\intro

root@tty1[/]#jobs
[1]+ Stopped            sleep 20
root@tty1[/]# bg 1
[1]+ sleep 20 &
[1]+ Done               sleep 20
root@tty1[/]#
```

Tenemos que, al haber transcurrido los 20 segundos, el proceso ha terminado estando aún en *background*, y se nos ha informado por pantalla. Podemos comprobar mediante `ps` que efectivamente no hay ningún proceso corriendo. En este punto, podemos parar un proceso y devolverlo a *foreground* mediante `fg`:

```
root@tty1[/]# man ls


<Ctrl+z>
[1]+ Stopped            man ls

root@tty1[/]# jobs
[1]+ Stopped            man ls
root@tty1[/]# bg 1
```

3.9. Activación y uso del ratón

El ratón es una herramienta extremadamente útil para trabajar en modo consola, ya que permite seleccionar un texto y pegarlo (así es como se han capturado todas las entradas y salidas para redactar el texto de este taller, por ejemplo).

ANOTACIONES

Por lo tanto, lo primero que haremos es configurar el ratón, y lo haremos con el programa más comúnmente utilizado, el `gpm`, el cual automáticamente ya corre en modo *background*. (Esto es debido a que gpm es un daemon, termino que se analiza con detalle en secciones posteriores.)

Hay muchos tipos de *ratones*, pero los más corrientes son los de tres botones conectados al ordenador vía puerto serie o vía puerto PS2. Si tenemos un ratón del primer tipo, ejecutaremos la línea siguiente (se presupone que el ratón está conectado en el primer puerto serie del ordenador, si no fuese así, soló habría que cambiar el *device* ttyS0 (`/dev/ttS0`) por ttyS1 (`/dev/ttS1`) en caso de estar en el segundo puerto, por ttyS2 para el tercer puerto o por ttyS3 para el cuarto puerto):

```
#gpm -m /dev/ttyS0 -t msc
```

Si lo que tenemos es un ratón conectado al puerto PS2, para activarlo se usará el mismo procedimiento anterior variando únicamente el *device* y el tipo de ratón:

```
#gpm -m /dev/psaux -t ps2
```

En principio, después de ejecutar una de las dos líneas de comandos anteriores, como *root,* moviendo el ratón deberíamos ver, en todos los terminales, el indicador de posicionamiento del ratón, y al ejecutar:

```
$roger@etseisa7:~$ ps aux | grep gpm
```

deberíamos obtener una respuesta del tipo

```
root 182 0.0 0.1 1552 524 ? S Mar11 0:00 /usr/sbin/gpm -m /dev/psaux -t ps2
```

conforme `gpm` está corriendo en *background*.

Si no fuese así, es que nuestro ratón no es estándar. En este caso deberemos leer con atención el `man` de `gpm` y ejecutar “`gpm -t`

ANOTACIONES

`help`" para tratar de identificar el tipo de ratón que se adapte al que tenemos.

Una vez configurado mediante el botón de la izquierda, podemos seleccionar la parte de texto que nos interese, y al pulsar el botón del medio pegaremos el contenido seleccionado en la posición actual del cursor. Si se está acostumbrado a usar el ratón para marcar el posicionamiento del cursor, puede que los resultados que obtengamos no sean precisamente los deseados, pero practicando un poco, enseguida nos acostumbraremos a este modo de operar.

Hay que destacar que el contenido del *búffer* del ratón se conserva al pasar de una tty a otra.

Actividad

10. Sacando provecho del hecho de que el contenido del *búffer* del ratón se mantiene entre terminales, abrir en una sesión de `vi` y en otra situarse en la raíz del sistema de ficheros, listar sus contenidos con detalle, y portar estos datos al editor de texto. Salvar los contenidos del fichero de texto y mediante `cat` comprobar que efectivamente hemos obtenido los resultados deseados.

3.10. Otras operaciones

Para finalizar este primer taller, ejecutaremos una serie de comandos para seguir el proceso de familiarización con el sistema e ir adquiriendo recursos para poder solucionar futuros problemas que puedan surgir.

Debemos informarnos acerca del sistema, el tipo de máquina sobre el que corre, el hardware que tengamos instalado, etc. (los retornos de los comandos que siguen serán, naturalmente, distintos para cada caso en particular):

ANOTACIONES

```
root@tty1[/]# uname -a
Linux Knoppix 2.4.20-xfs #1 SMP Die Dez 10 20:07:25 CET 2002 i686 AMD Athlon(TM) XP
                                                     2100+ AuthenticAMD GNU/Linux

root@tty1[/]# cat/proc/cpuinfo
processor      : 0
vendor_id      : AuthenticAMD
cpu family     : 6
model          : 8
model name     : AMD Athlon(TM) XP 2100+
stepping       : 1
cpu MHz        : 1732.890
cache size     : 256 KB
fdiv_bug       : no
hlt_bug        : no
f00f_bug       : no
coma_bug       : no
fpu            : yes
fpu_exception  : yes
cpuid level    : 1
wp             : yes
flags          : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca
                 cmov pat pse36 mmx fxsr sse syscall mmxext 3dnowext 3dnow
bogomips       : 3460.30

root@tty1[/]# lspci
00:00.0 Host bridge: VIA Technologies, Inc. VT8367 [KT266]
00:01.0 PCI bridge: VIA Technologies, Inc. VT8367 [KT266 AGP]
00:05.0 Multimedia audio controller: C-Media Electronics Inc CM8738 (rev 10)
00:06.0 RAID bus controller: Promise Technology, Inc. PDC20276 IDE (rev 01)
00:07.0 FireWire (IEEE 1394): Texas Instruments TSB43AB22 1394a-2000 Controller
00:09.0 USB Controller: VIA Technologies, Inc. USB 6(rev 50)
00:09.1 USB Controller: VIA Technologies, Inc. USB (rev 50)
00:09.2 USB Controller: VIA Technologies, Inc. USB 2.0 (rev 51)
00:0d.0 SCSI storage controller: Artop Electronic Corp AEC6712D SCSI (rev 01)
00:0f.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8029(AS)
00:11.0 ISA bridge: VIA Technologies, Inc. VT8233A ISA Bridge
00:11.1 IDE interface: VIA Technologies, Inc. Bus Master IDE (rev 06)
00:11.2 USB Controller: VIA Technologies, Inc. USB (rev 23)
00:11.3 USB Controller: VIA Technologies, Inc. USB (rev 23)
01:00.0 VGA compatible controller: ATI Technologies Inc Radeon 7500 QW

root@tty1[/]# df
Filesystem     1K-blocks     Used         Available     Use%      Mounted on
/dev/root      1971          1516         455           77%       /
/dev/CD-ROM    715744        715744       0             100%      /CD-ROM
/dev/cloop     1837536       1837536      0             100%      /KNOPPIX
/dev/shm       407928        40           407888        1%        /ramdisk
/dev/fd0       1424          56           1368          4%        /mnt/auto/floppy
```

Actividad

11. En el directorio virtual `/proc/` y en sus subdirectorios encontraremos mucha información relativa a nuestro sistema en ficheros de texto. Situarse en este directorio, y mediante `more` o `cat` explorar la información de estos ficheros.

3.11. Conclusión

Éste ha sido nuestro primer contacto con un entorno UNIX. Ha servido un poco para romper el mito de que trabajar en este tipo de sistemas operativos es complicado y difícil. Es más, ha servido para que empecemos a intuir su potencia y versatilidad. Hemos aprendido a movernos por su sistema de ficheros y a realizar operaciones con éstos, a buscar información, etc. Todo lo que hemos aprendido hasta ahora nos será de gran utilidad de aquí en adelante, y nos servirá de base para ampliar nuestros conocimientos.

Por otra parte, la gran potencia de autodetección y configuración de hardware de KNOPPIX hace que éste nos pueda ser muy útil a la hora de instalar Debian en el próximo taller si surgieran problemas de hardware. Siempre podremos volver a arrancar KNOPPIX y estudiar cómo ha solucionado éste el problema (básicamente estudiando los archivos de configuración).

Como última actividad del taller, podemos volver a arrancar KNOPPIX en modo gráfico, es decir, sin pasar el parámetro 2 al iniciarlo (una manera de hacerlo puede ser pasando los parámetros ”knoppix lang=es wheelmouse”).

ANOTACIONES

4. Instalación de GNU/Linux

4.1. Introducción

En este capítulo veremos los pasos esenciales que se siguen en la mayoría de procesos de instalación de GNU/Linux. Aunque cada distribución tiene su propio entorno de instalación, en todas ellas deben existir unos pasos básicos. En este capítulo veremos estos pasos, explicando los conceptos necesarios para superarlos correctamente. Es importante que antes de instalar un nuevo sistema operativo conozcamos adecuadamente los componentes principales que tenemos instalados en nuestro ordenador para poder configurarlo todo adecuadamente, aun cuando la distribución que utilicemos incorpore detección de hardware.

Es posible que en un solo disco duro tengamos instalados dos o más sistemas operativos totalmente independientes. Aunque el proceso de instalación de otro operativo en el mismo disco (o en otro instalado en el ordenador) no debería interferir con las particiones de los demás, es aconsejable hacer copias de seguridad de todos nuestros documentos. Sin embargo, si seguimos adecuadamente los pasos que detallaremos a continuación es prácticamente imposible perder información, siempre es recomendable la prudencia y realizar copias de los archivos que realmente nos importan.

Nota

Antes de empezar el proceso de instalación es aconsejable conocer la marca y modelo de la tarjeta gráfica y de sonido que tenemos instalada, la tarjeta de red, la marca, tipo y características del monitor, y cualquier otro hardware especial que tengamos. Generalmente, para la placa base, la CPU y la memoria RAM, no suele ser necesario conocer sus características. Para obtener esta información, podemos recurrir a los manuales entregados en la compra del ordenador o, si tenemos otro sistema operativo instalado, recurrir a él para este fin (en sistemas Windows™ lo podemos conseguir a partir del panel de control).

4.2. Arrancando

Generalmente, todas las distribuciones de GNU/Linux proporcionan algún tipo de medio para el arranque del proceso de instalación. Actualmente se suele proporcionar un CD o DVD de arranque o, en caso de que nuestro ordenador no tenga el lector correspondiente o queramos realizar una instalación remota, se proporciona un disquete de arranque. Para empezar con el proceso, debemos introducir el medio incluido en su dispositivo y arrancar el ordenador, configurando su BIOS o EFI para que arranque desde la unidad deseada.

El primer paso del proceso suele ser escoger qué tipo de instalación queremos realizar. Generalmente, la elección suele ser para identificar qué tipo de usuario está instalando el sistema para, según éste, proporcionarle más o menos información y dejarle configurar con más precisión el sistema o no. Si tenemos los conocimientos suficientes, es recomendable hacer la instalación en modo experto (o similar) para poder adaptar más el sistema a nuestras necesidades específicas. En algunas distribuciones, este primer paso sirve para seleccionar la versión del núcleo que queremos instalar o para configurar el proceso de instalación en modo gráfico, texto, etc. Es imprescindible leer antentamente la información que se nos proporciona en esta primera fase para poder elegir adecuadamente lo que más se ajuste al destino que queramos darle al sistema.

Contenido complementario

Algunas distribuciones de GNU/Linux nos permiten realizar la instalación desde cualquier medio: CD, DVD, FTP, HTTP, disco duro, NFS, etc. También existen métodos de instalación que permiten arrancar el proceso desde otros sistemas operativos.

Seguidamente, la mayoría de distribuciones nos dejan elegir el tipo de teclado (o configuración similar) que queremos utilizar. Si bien los hay de muchas clases diferentes, los más frecuentes son los `qwerty` (los primeros caracteres empezando por arriba y la izquierda del teclado), con lo cual deberíamos seleccionar qwerty/es (Spain).

4.3. Fraccionando el disco

Fraccionar el disco duro es una de las partes más críticas de todo el proceso. Este paso significa dividir el disco duro en varias secciones, por lo que cada una de ellas se toma como unidad independiente. Si ya tenemos un sistema operativo instalado en nuestro ordenador, el disco estará fraccionado en una o varias particiones, pero si el disco es nuevo, generalmente no.

Contenido complementario

Aunque hay formas de instalar GNU/Linux utilizando el sistema de ficheros de otro operativo, no es recomendable utilizar este tipo de instalación porque el rendimiento del sistema baja considerablemente.

Para instalar GNU/Linux debemos disponer de, al menos, una partición para él. El hecho de que al modificar el tamaño de una partición debemos eliminarla y crearla de nuevo implica perder toda la información que tenemos en ella. Por éste y otros motivos, existen programas que nos permiten modificar el tamaño de las mismas sin tener que eliminarlas. El `fips` es un programa con licencia GPL que nos permite redimensionar nuestras particiones formateadas con FAT (para sistemas Windows™ de la rama no

ANOTACIONES

NT) sin perder la información de las mismas. También existen otros programas comerciales que nos permiten efectuar este tipo de operación con cualquier otro sistema de ficheros, de forma que si no queremos perder la información de nuestros sistemas, deberemos utilizar alguno de ellos antes de empezar con todo el proceso.

Es recomendable que GNU/Linux utilice dos particiones en el disco duro. Una servirá para guardar los ficheros del sistema y la otra para el ***swap***. El *swap* es una zona de intercambio entre la memoria RAM del ordenador y el disco duro. Sirve cuando el sistema operativo tiene toda la memoria RAM ocupada y los programas en ejecución piden más. Es en este momento cuando se empieza a utilizar el *swap* para guardar zonas de RAM que no se están utilizando, intercambiándolas para que las aplicaciones no se queden sin memoria disponible. También es posible prescindir de *swap*, pero no es recomendable porque el sistema no podrá gestionar tan adecuadamente sus recursos; además, al utilizar simultáneamente muchas aplicaciones, éstas se quedarán sin memoria con más facilidad. Aunque el tamaño del *swap* puede ser tan grande como queramos, se recomienda que éste sea el doble de la RAM instalada en el ordenador si tenemos 64MB o menos, y igual si tenemos más. Estos cálculos están basados en pruebas de rendimiento del sistema que nos demuestran que llega un punto en el que, si las aplicaciones necesitan utilizar demasiada memoria *swap*, el rendimiento se decrementa mucho, haciendo que el sistema quede prácticamente saturado (para ver qué cantidad de memoria RAM y *swap* se está utilizando, el sistema nos proporciona el comando `free`).

Existen varias aplicaciones para fragmentar el disco. Una de las primeras que apareció fue el `fdisk`, aunque actualmente existen otras, como `cfdisk`, `diskDruid`, etc. En algunos procesos de instalación se puede escoger cuál queremos utilizar, aunque todas permiten hacer exactamente lo mismo cambiando, eso sí, la presentación, el entorno, etc. La forma como GNU/Linux identifica los discos es `/dev/hdX` para los discos IDE y `/dev/sdX` para los SCSI, donde en ambos casos la "`X`" es una letra, correspondiente con el disco al que nos queramos referir de la siguiente manera:

dispositivo	significado
/dev/hda	Maestro del primer canal IDE

Contenido complementario

Si bien con una o dos particiones es suficiente para poder instalar GNU/Linux, es interesante dividir el disco en más fragmentos y situar ciertos directorios en diferentes unidades para poder hacer una gestión más eficiente de los recursos, evitar caídas del sistema por saturación de disco, etc. De todos modos, estos aspectos los dejaremos para cursos más avanzados de administración.

Contenido complementario

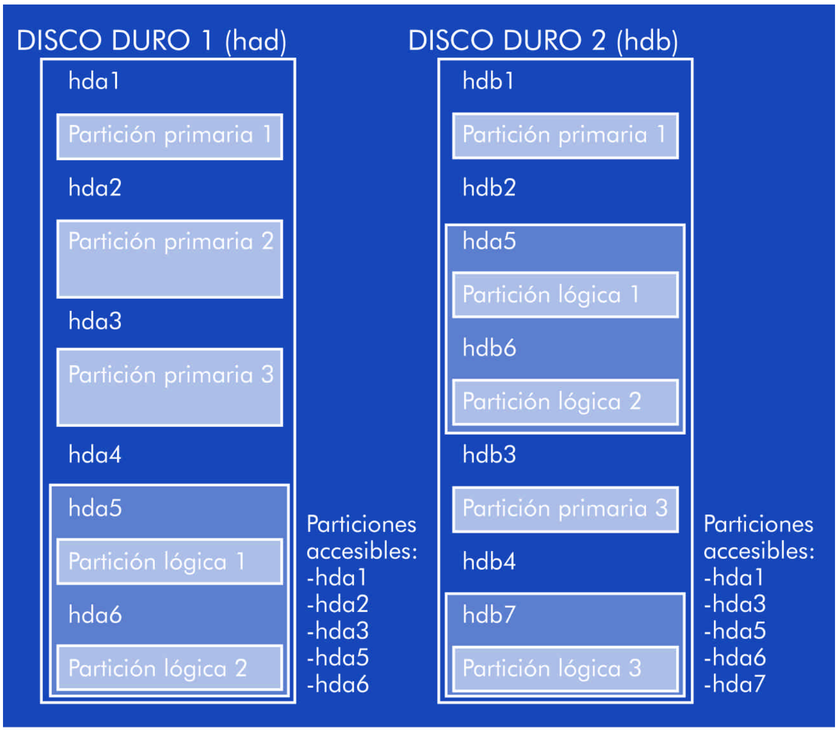
Para los ordenadores personales existen dos clases de discos duros: los IDE y los SCSI. Las controladoras IDE (*Integrated Drive Electronics*) son las que llevan la mayoría de placas base. Con este controlador podemos conectar dos dispositivos compatibles (discos, CD, etc.) en cada uno de los dos canales. Las controladoras SCSI (*Small/Smart Computer System Interface*) permiten hasta 8 dispositivos y tienen tasas de transferencia más altas, aunque su precio también es bastante más elevado que los IDE.

Contenido complementario

Para guardar la información de particiones y el programa de arranque, los discos tienen una zona de datos reservada llamada MBR (*Master Boot Record*).

dispositivo	significado
/dev/hdb	Esclavo del primer canal IDE
/dev/hdc	Maestro del segundo canal IDE
/dev/hdd	Esclavo del segundo canal IDE
/dev/sda	Primer disco de la controladora SCSI
/dev/sdb	Segundo disco de la controladora SCSI
...	

Si tenemos más de un disco en nuestro ordenador, antes de entrar en el programa de fraccionamiento podremos elegir sobre cuál de ellos operar. Cuando creamos una partición se nos preguntará si debe ser primaria o lógica. En un disco duro podemos tener hasta cuatro particiones primarias y hasta 64 lógicas. Si no necesitamos más de 4 particiones, podemos escoger cualquiera de los dos tipos. Si necesitamos más, deberemos tener en cuenta que las lógicas se sitúan dentro de una primaria (hasta un máximo de 16 para cada una), de forma que no podemos tener 4 particiones primarias creadas y después añadir otras lógicas. En este caso, deberíamos crear 3 primarias y hasta 16 lógicas en la cuarta partición primaria. En la siguiente figura podemos ver un ejemplo de manera gráfica:



ANOTACIONES

Cuando creamos una partición, debemos indicar qué sistema de ficheros utilizaremos para ésta (`Linux ext2`, `Linux ext3` o `Linux swap`). Una vez realizadas las particiones, guardaremos la configuración y debemos indicar al proceso de instalación dónde queremos situar la raíz del sistema de ficheros (***root filesystem***) y el *swap* del sistema. Una vez efectuados estos pasos, ya podremos continuar con el resto del proceso de instalación.

Contenido complementario

De todas las particiones de un disco duro podemos elegir una para que sea la activa. Este *flag* sirve para indicar a la BIOS o la EFI del sistema cuál es la partición que debe iniciar si en la MBR no encuentra ningún programa de arranque.

4.4. Instalación de módulos

Los módulos del núcleo son partes de software especializadas en alguna tarea concreta. Al haber toda clase de dispositivos diferentes y decenas de funciones para múltiples sistemas de ficheros, operaciones de gestión en red, etc., se decidió no incluir de forma estándar todos estos *drivers* y funciones. En las primeras versiones de Linux, cuando el núcleo que teníamos no incluía alguna función, debíamos recompilarlo completamente y generar uno nuevo adaptado a nuestras necesidades. Con la incorporación de los módulos, todo este proceso no es necesario y podemos seleccionar directamente las funciones que necesitamos.

Generalmente, las distribuciones ya incorporan centenares de módulos diferentes para el núcleo Linux. Se suelen organizar en diferentes categorías para facilitar su localización y, normalmente, con el mismo nombre del módulo o la breve descripción que se facilita con el mismo ya sabremos para qué dispositivo está diseñado o qué función realiza. Por esta razón decíamos anteriormente que era importante conocer el hardware de nuestro ordenador: en este paso podremos elegir con más precisión los módulos que necesitamos. En algunos de ellos es obligatorio pasar algún parámetro especial (como la dirección de E/S, la interrupción que utiliza el dispositivo, etc.), información que podemos obtener por medio de la BIOS o EFI del sistema o por los procedimientos que anteriormente comentábamos.

También es cierto que si el proceso de instalación lleva algún tipo de programa para el reconocimiento automático de hardware, todo el proceso de selección y carga de módulos no es necesario porque ya se realiza de forma automática. Sin embargo, es posible que no se

ANOTACIONES

detecte adecuadamente alguno de los dispositivos que tengamos instalados, en cuyo caso sí que debemos incluir manualmente el módulo correspondiente.

Contenido complementario

El fichero de configuración de los módulos se suele encontrar en `/etc/modules`.

Si en el momento de instalar el sistema nos olvidamos de incluir algún módulo (o instalamos algún nuevo dispositivo) siempre podemos recurrir a los comandos `insmod` (para agregar un nuevo módulo), `lsmod` (para listar los instalados), `rmmod` (para eliminar alguno) y `modprobe` (para probar alguno y, si funciona correctamente, incluirlo en el núcleo). Todos estos módulos no son más que ficheros binarios que acostumbramos a encontrar en el directorio `/lib/modules/` del sistema.

Si tenemos problemas para descubrir qué hardware tenemos instalado en el ordenador, una utilidad muy práctica que se suele incluir en las distribuciones es `discover`. Con ésta, podremos saber exactamente qué componentes tenemos y qué módulos les corresponden.

4.5. Configuración básica de la red

Después de configurar los módulos que se incluirán en el núcleo del sistema operativo deberemos configurar la red (si tenemos la tarjeta necesaria). Aunque en este documento no entraremos en detalles sobre redes de computadores, daremos las ideas necesarias para poder realizar este paso sin complicaciones.

Contenido complementario

Una dirección IP es la identificación de un ordenador dentro de una red cuando utilizamos el protocolo TCP/IP, el que más se utiliza en Internet.

Lo primero que se pedirá es el nombre que queremos darle al sistema. Este nombre servirá para referirnos a él sin tener que recordar siempre su dirección IP. Una vez entrado el nombre, se suele pedir si en nuestra red utilizamos un mecanismo llamado DHCP o BOOTP, que consiste en tener un servidor especial que se encarga de asignar automáticamente las IP a los ordenadores que arrancan. Si utilizamos este mecanismo, debemos indicarlo y si no, se nos preguntará por la IP y máscara de nuestro ordenador. Si no conocemos estos datos, debemos dirigirnos al administrador de nuestra red. Seguidamente debemos introducir la IP del *gateway* de nuestra red. El *gateway* es un dispositivo u ordenador que actúa de puente entre

nuestra red local e Internet (si no tenemos ningún dispositivo de este tipo, podemos dejar en blanco este campo). A continuación, debemos especificar el (o los) servidores de nombres que utilizamos. Un servidor de nombres (servidor DNS) es una máquina que nos proporciona la equivalencia entre un nombre y una dirección IP. Si no sabemos cuál usamos, también deberemos recurrir al administrador de la red.

Si estamos en una red local, debemos consultar al administrador de la misma para que nos proporcione toda la información necesaria al respecto. Si tenemos otro operativo instalado en el ordenador, también podemos acceder a su configuración para obtenerla. Sin embargo, en ningún caso nos podemos inventar estos valores porque lo más probable es que no se configure adecuadamente y provoquemos problemas en la red local.

4.6. Sistema de arranque

Una vez configurados todos estos aspectos, debemos instalar un pequeño programa en el disco duro para que en el proceso de arranque del ordenador podamos elegir qué sistema operativo de los que tenemos instalados queremos arrancar. Aunque sólo hayamos instalado GNU/Linux, también tendremos que instalar este programa, ya que el sistema de arranque del mismo lo necesita.

Existen varias aplicaciones para realizar este proceso. Las más usuales son el Lilo (*LInux LOader*) y el Grub (GNU *GRand Unified Bootloader*). Lo único que hacen estas aplicaciones es iniciar el proceso de carga y ejecución del núcleo del sistema operativo que le indiquemos. Generalmente, todas las distribuciones detectan si tenemos algún otro sistema operativo instalado en los discos duros y nos configuran automáticamente el sistema de arranque. Lo único que debemos tener en cuenta es que habrá que situar este programa correctamente para que se ejecute al arrancar el ordenador. Normalmente se suele poner en la MBR del disco maestro del primer canal IDE o SCSI, que es el primer sitio que la BIOS o EFI del ordenador inspecciona buscando un programa de estas características.

ANOTACIONES

4.7. Elección de paquetes

La mayoría de procesos de instalación incluyen dos formas de seleccionar los programas que instalaremos en el sistema: básico o experto. Con el proceso de selección básico generalmente se agrupan los paquetes disponibles por grandes grupos de programas: administración, desarrollo de software, ofimática, matemáticas, etc. Si todavía no conocemos exactamente los programas que vamos a utilizar, este tipo de instalación es el más adecuado porque podremos escoger, de manera muy general y sin entrar en detalles, qué tipo de programas utilizamos.

Contenido complementario

Un paquete está formado por uno o varios programas/librerías/ relacionados entre sí, que se agrupan para formar un sólo bloque. La mayoría de distribuciones incluyen utilidades para el manejo de paquetes (instalación, eliminación, configuración, etc.). Esta clase de organización es muy útil porque al necesitar un programa/utilidad/etc. podemos instalarlo todo a la vez.

Cuando conozcamos un poco más el sistema y sepamos qué es exactamente lo que vamos a utilizar, es mejor la selección experta de paquetes. Con este tipo de selección podremos ajustar mucho mejor qué programas necesitamos, ahorrándonos espacio en el disco y evitando que el sistema cargue más programas de los necesarios. Al instalar un sistema para un uso específico (servidor http, cvs, etc.) es muy recomendable escoger sólo aquellos programas que realmente utilizaremos para evitar problemas de seguridad (cuantas menos puertas dejemos abiertas al exterior, más seguro será el sistema).

Contenido complementario

RedHat también ha adoptado un sistema de descarga/ actualización del mismo estilo que el `apt` de Debian.

Algunas distribuciones también admiten la posibilidad de obtener los paquetes desde ubicaciones diferentes, como uno o varios CD, desde servidores en Internet, etc. Debian GNU/Linux fue la primera en tener un sistema de este tipo, llamado `apt`. Este tipo de gestión es muy útil porque nos proporciona mucha flexibilidad, y nos mantiene informados de las últimas correcciones y actualizaciones de los programas.

4.8. Otros aspectos

Debemos saber qué hacer si hemos tenido algún problema con la instalación o en algún momento el sistema de arranque no nos deja cargar el sistema operativo. Generalmente, en el proceso de instalación existe algún paso donde se nos pregunta si queremos crear un disquete de recuperación. Siempre es muy recomendable generar este disquete, ya que nos permite cargar el operativo y acceder al sistema de ficheros para arreglar lo que haga falta.

En algunas distribuciones, con el mismo CD de arranque podremos hacer lo mismo. En el caso de Debian, por ejemplo, con el mismo proceso de arranque se incluye una consola (apretando CTRL+F2 podemos acceder a ella) con los comandos básicos para que podamos realizar operaciones esenciales de recuperación. De todos modos, si nos encontramos con algún grave inconveniente que no nos permite arrancar el sistema correctamente y queremos todas las herramientas usuales, también podemos arrancar con un CD-live de Knoppix o cualquier otra distribución y montar la unidad donde tengamos instalado el sistema para arreglar la disfunción. Lo que sí que es importante es disponer de alguna de estas herramientas y haberla probado antes de que ocurra algún problema serio para estar siempre preparados.

La mayoría de procesos de instalación utilizan una aplicación denominada `bootstrap`, que también podemos instalar en el sistema con el paquete correspondiente. Con ella podríamos crearnos nuestro propio proceso de instalación o ver cómo realiza realmente el sistema estas operaciones de instalación y configuración. La mayoría de ellas las podemos reproducir con el programa `base-config` o alguno de los otros que él mismo llama (más información en su manual), aunque siempre podemos recurrir a los mismos ficheros de configuración del operativo para realizarlo manualmente.

ANOTACIONES

5. Taller de instalación de Debian Woody

5.1. Introducción

Si el primer taller nos servía para dar nuestros primeros pasos sobre un sistema tipo UNIX, y para ello utilizábamos una distribución que no dejase rastro en nuestro PC, pues se arrancaba y se ejecutaba desde CD-ROM, éste debe servirnos para aprender a instalar un sistema básico GNU/Linux en nuestro ordenador. La distribución escogida para el desarrollo del taller ha sido la Debian 3.0r1 (las versiones 3.0 también se conocen con el nombre de Debian Woody).

La decisión de escoger Debian frente a RedHat para el desarrollo de este taller, y de los dos restantes, no ha sido nada fácil, pues RedHat ofrece productos comerciales que tienen muy buena acogida en el mundo empresarial, y en general se considera mucho más sencillo instalar una RedHat que una Debian. El hecho de que se haya optado por Debian se debe principalmente: en primer lugar, a que esta distribución es la que sigue más fielmente la filosofía GNU/Linux y todo se hace gracias al trabajo voluntario de miles de personas; en segundo lugar, porque probablemente es una de las distribuciones que, tanto a la hora de instalar como de mantener, deja más libertad a los usuarios; y como último punto fundamental, por su sistema de paquetes, que probablemente es el más consistente que existe actualmente.

Se ha optado, pues, por Debian; no porque se considere mejor que RedHat, (seguro que existen tantas opiniones al respecto como usuarios de ambas distribuciones), sino porque simplemente se ha creído que su flexibilidad la convierte en una distribución muy apta para fines didácticos. Pero siendo conscientes del fuerte arraigo que tiene RedHat, se ha dedicado un apéndice a esta distribución, para que el lector pueda conocerla también y pueda formarse su propia opinión.

ANOTACIONES

Lectura complementaria

http://www.debian.org/doc/

http://www.debian.org/distrib/cd

http://www.debian.org/distrib/netinst

Tampoco hay que olvidar que, aparte de estas dos distribuciones, existen muchas más. Pero creemos que, cuando ya se ha tenido un primer contacto con alguna de ellas, es tarea de cada usuario ir probando distintas distribuciones e ir forjándose una opinión propia de cuál es la que se adapta mejor a sus necesidades o exigencias. Por este motivo, una vez más animamos a hacer todo tipo de experimentos y pruebas, y que cada cual dirija sus esfuerzos hacia donde crea más interesante. GNU/Linux no es un sistema cerrado, sino todo lo contrario, GNU/Linux es sinónimo de libertad, y por este motivo la intención básica de este módulo es, en general, que una vez terminado se hayan sentado las bases de conocimiento necesarias para que se pueda ejercer sin ningún tipo de impedimento esta libertad, con el beneficio de todas sus ventajas y asumiendo todas sus consecuencias.

Actividad

12. Ya que se va a instalar Debian, es recomendable visitar su página web y familiarizarse un poco con sus contenidos. Así pues, se propone visitar la web `http://www.debian.org` y sus subapartados. Una vez más se insta al lector a dejarse guiar por su curiosidad y, en esta ocasión, a seguir los enlaces que le parezcan interesantes.

5.1.1. Sistemas de instalación

Por sistema de instalación se entiende qué recursos o dispositivos se van a utilizar para hacer la instalación del sistema. Actualmente, casi todas las distribuciones ofrecen diversas posibilidades para realizarla, pero esencialmente se puede distinguir entre dos sistemas de instalación: mediante CD-ROM o por red. Además, es posible, en general, combinar distintos sistemas.

Debian ofrece tres tipos de instalación: mediante un juego de CD-ROM, con un único CD-ROM, el cual contiene el software de instalación y los paquetes básicos, dejando que el resto de la instalación se haga por red, y exclusivamente por red. Aunque, obviamente, si se opta por este último sistema, hará falta algún tipo de soporte inicial para arrancar el sistema de instalación (normalmente un juego de

ANOTACIONES

disquetes, entre dos y diecisiete, dependiendo del tipo y sistema de instalación escogidos, o un CD-ROM, del propio juego de CD de la distribución, aunque también es posible usar otros métodos, como DHCP para hacer una instalación completamente remota).

El grado de interacción que requiere una instalación también depende del sistema y del tipo de instalación escogidos. Tal como es de esperar, cada uno de estos sistemas tiene sus ventajas y sus inconvenientes: mientras que una instalación estándar nos permite ir configurando paso a paso, hecho que nos será extremadamente útil para adecuar el sistema a nuestras necesidades y posibilidades, un sistema de instalación totalmente automático requiere de unas infraestructuras y de unos conocimientos más avanzados, y por tanto de una inversión, tanto en tiempo como en dinero, que queda justificada sólo si el número de sistemas a montar es muy grande (por ejemplo, se podría plantear la implementación de este tipo de instalación en un departamento donde convivieran ordenadores destinados a uso personal, con otros destinados a la paralelización, y donde su número aumenta a menudo).

A la hora de escoger un sistema y tipo de instalación, debemos considerar, pues, diversos factores, como son: ¿cuántas instalaciones vamos a realizar?, ¿cuántas instalaciones distintas vamos a realizar?, ¿qué grado de experiencia tenemos?, etc. El hecho de ser ésta nuestra primera instalación nos lleva inmediatamente al tipo de instalación más interactiva y más utilizada: la instalación interactiva estándar. Ahora sólo falta por determinar el sistema de instalación, y esto dependerá fundamentalmente de si disponemos de conexión a Internet y de la velocidad de la misma. Obviamente, si no disponemos de conexión a Internet o la velocidad de acceso que tenemos es muy baja (como la que ofrecen los módems estándares) no tenemos más opción que escoger una instalación basada en un juego de CD-ROM; si por el contrario disponemos de una velocidad de acceso a Internet medianamente aceptable (como la que pueden ofrecer las líneas basadas en tecnología ADSL) o alta (conexión directa a Internet via *gateway*), la mejor opción será decantarse por una instalación por red.

Una instalación efectuada por red supone muchas ventajas sobre una efectuada mediante CD-ROM, y especialmente en Debian, ya que ello nos permitirá instalar las últimas versiones disponibles de los paquetes, y actualizar todos los paquetes instalados en el sistema

ANOTACIONES

será tan simple como ejecutar una sola instrucción. Pero este hecho no debe hacernos abandonar la instalación utilizando CD-ROM, si, como hemos dicho, no disponemos de conexión a Internet o la que tenemos es muy lenta (no hay que subestimar tampoco los paquetes que contienen estos CD-ROM, ya que Debian se caracteriza por ser una distribución donde sólo se incluyen paquetes que han sido probados exhaustivamente).

5.1.2. Tipos de paquetes

Lectura complementaria

http://www.debian.org/distrib/packages

http://www.debian.org/social contract#guidelines

A continuación nos centraremos en Debian y en su sistema de paquetes. Un paquete de Debian es identificable por su extensión `.deb`. La distribución Debian diferencia sus paquetes en cuatro clases diferentes. Los paquetes propios de la distribución están en la clase main, mientras que las clases contrib, non-free y non-US los provee la organización Debian para el beneficio de sus usuarios:

main. Paquetes que cumplen con las Debian Free Software Guidelines, es decir, que se garantiza su uso y redistribución libre, tanto de todos los binarios que los componen como de su código fuente completo.

contrib. Paquetes que, aun siendo libres, y por tanto aparte de los binarios –también tienen disponible su código fuente–, dependen de otros paquetes que no lo son.

non-free. Paquetes que, aun pudiendo no costar dinero, están bajo condiciones onerosas que restringen de alguna forma su uso o redistribución.

non-US/main non-US/non-free. Paquetes que no pueden ser exportados fuera de EE.UU., por contener software de cifrado o software que puede afectar a asuntos relacionados con patentes.

5.1.3. Estado de desarrollo de los paquetes

El estado de desarrollo de los paquetes marcará el tipo de distribución que instalemos. Así pues, se habla de tres tipos de distribución:

stable. Distribución de los paquetes con la versión oficial más reciente de la distribución. Consta de softwre estable y bien probado, y

ANOTACIONES

cambia sólo al incorporar correcciones importantes de seguridad o de usabilidad.

testing. Distribución que contiene los paquetes que se espera que formen parte de la próxima distribución estable. Hay una serie de requisitos muy estrictos que debe cumplir cada paquete antes de dejar de ser *unstable* para pasar a ser *testing*.

Lectura complementaria

http://www.debian.org/releases/testing/

unstable. En esta distribución son los paquetes más recientes de Debian y en consecuencia los menos probados. Por esta razón, pueden contener problemas suficientemente graves como para afectar a la estabilidad del sistema.

Aunque todos los paquetes tienen sus propias dependencias no hay ningún problema en mezclar paquetes de distintas distribuciones. Apt-pinning (man apt-pinning) facilita mucho esta tarea. Aún así, en sistemas críticos es recomendable utilizar solamente paquetes de la distribución estable, los más fiables.

Nota

Es posible incorporar paquetes a una instalación basada en una distribución procedentes de otra distribución, esto por lo general implica serios problemas de compatibilidades y es por este motivo por lo que sólo se recurre a esta práctica en casos puntuales.

5.2. Instalación de Debian Woody

Como se ha dicho, en esencia Debian presenta tres tipos de instalación: vía juego de CD-ROM, vía CD-ROM mínimo, e instalación por red. Si se dominan la primera y la última, utilizar la segunda es trivial. Así pues, empezaremos por instalar nuestro primer sistema mediante el juego de CD-ROM facilitados paso a paso, y, una vez terminado, analizaremos las diferencias entre este proceso y el de instalación por red.

Hay que añadir que el método de instalación vía CD-ROM mínimo, si bien está admitido como tal por parte de Debian, las imágenes de estos CD-ROM no son oficiales. Estas imágenes son creadas por cuenta propia, generalmente por gente perteneciente a la Debian *people*, y sólo están referenciadas en la página oficial de Debian, en el apartado de instalación mediante CD-ROM mínimo. Aun así, en general este sistema de instalación es muy práctico, y ha sido probado con buenos resultados, con imágenes de distintas fuentes, en diversos ordenadores.

Lectura complementaria

http://db.debian.org/

http://www.debian.org/CD/netinst/

ANOTACIONES

5.2.1. *Flavours* de Debian Woody

Se conoce con el nombre de *flavors* o 'sabores', a los distintos *kernels* precompilados, destinados a soportar distintos tipos de hardware de las distribuciones. Debian Woody presenta *cuatro flavors* distintos:

vanilla. Este *kernel* está basado en la serie 2.2, en el cual se han incluido muchos *drivers* destinados a soportar hardware antiguo (como pueden ser los dispositivos ISA), además de dar soporte al puerto USB.

compact. Este *kernel* está basado en la serie 2.2 y principalmente orientado a dar soporte a dispositivos PCI y sistemas IDE y SCSI.

idepci. Este *kernel* está basado en la serie 2.2 y está concebido para dar soporte al mayor número de hardware posible. Por esta razón se trata de un *kernel* bastante grande y poco optimizado.

bf24. *Kernel* basado en la serie 2.4 con soporte para ext3 y ReiserFS orientado a dar soporte a hardware nuevo, como pueden ser los teclados USB. También da soporte a dispositivos SCSI e IDE ATA-100.

En este taller se hará la instalación utilizando el *flavor* **bf24**, ya que la experiencia dice que da muy buenos resultados, aparte de usar un *kernel* basado en la serie 2.4. Si con este *flavor* se diera el caso de que durante el proceso de arranque el ordenador se quedara colgado o notáramos comportamientos extraños, como podría ser el no reconocimiento de disco duro, deberíamos cambiar de *flavour,* intentando hallar el que se adaptara mejor a nuestras necesidades, o el que simplemente nos dejase arrancar sin problemas; en este caso la primera opción que debería probarse es la **idepci**.

5.2.2. CD-ROM de Debian Woody y sus distintos *flavours*

De los siete CD-ROM que componen Debian Woody, cinco llevan asociados uno o más *flavours*, y por tanto se puede arrancar el sistema de instalación desde ellos. La relación entre CD y *flavours* es la siguiente:

CD1. Este CD-ROM es "multiboot", es decir, al arrancar se le puede pasar como parámetro, el *flavor* deseado entre otros. Si no se le

ANOTACIONES

pasa ningún parámetro, transcurridos unos segundos arrancará como idepci.

CD2. vanilla.

CD3. compact.

CD4. idepci.

CD5. bf24.

Así pues, para inicializar nuestra instalación bastará con "bootar" el ordenador desde la unidad de CD-ROM con el CD1 y teclear `bf24` o con el CD5 y pulsar INTRO.

5.2.3. Installing Debian GNU/Linux 3.0 For Intel x86

El documento básico que nos proporciona Debian para su instalación es el *Installing Debian GNU/Linux 3.0 For Intel x86*, incluido en el CD1 (/install/doc/install .en .html) disponible en distintos idiomas, entre ellos el castellano (/install/doc/es/install.es.html). Es recomendable leer este documento y tener una copia del mismo a mano por si surgiera algún problema, como podría ser no disponer de una unidad de CD-ROM *bootable*.

5.3. Instalación de Debian Woody mediante CD-ROM

Esta instalación se hará en un PC estándar sobre un disco duro IDE conectado como máster sobre el puerto IED primario estándar. Si se dispone de otro tipo de conectores en la placa base más rápidos (ATA-100 o ATA-133), probablemente también sean soportados por Debian, pero en general se requiere de configuraciones especiales y, en consecuencia, quedan fuera de este taller. Por lo que se refiere a controladoras y discos SCSI, si éstos son medianamente estándares, serán detectados sin ningún problema durante el proceso de arranque.

ANOTACIONES

5.3.1. Antes de empezar la instalación

Antes de empezar la instalación propiamente dicha, habrá que cerciorarse de que disponemos de un espacio mínimo en nuestro disco duro donde realizarla (se recomienda disponer, como mínimo, de entre dos y tres gigabytes de espacio libre). Si éste es nuevo, podemos empezar directamente con la instalación, a pesar de que pensemos instalar también otro sistema operativo en él (bastará con reservar el espacio que consideremos para éste con el tipo de partición que requiera).

Si disponemos de un espacio que previamente habíamos reservado –pues ya teníamos en mente instalar GNU/Linux– o tenemos una partición de cualquier otro sistema operativo donde deseamos instalarlo, también podemos proseguir con la instalación, es decir, arrancar desde el CD-ROM.

Si por el contrario tenemos todo el disco duro ocupado y con una sola partición (cosa muy poco recomendable, pues en general esto hace disminuir sensiblemente el rendimiento de cualquier sistema operativo, y en especial aquellos con sistemas de ficheros poco consistentes), debemos liberar espacio para poder instalar Debian Woody. La dificultad de realizar esta operación dependerá estrictamente de qué sistema de ficheros sea el que contenga dicha partición.

Probablemente, el sistema operativo en cuestión sea de la familia de productos de Microsoft™; si el sistema de ficheros es de tipo FAT o FAT32 (utilizados por MSDOS™, Windows95™ y Windows98™) el problema es relativamente sencillo de solventar, ya que con la misma distribución se nos facilita una aplicación (`fips20.exe` en el CD1 (/tools/fips20.zip)) que nos asistirá en la repartición del disco duro y en la creación de espacio para instalar GNU/Linux (es muy recomendable que antes de hacer correr `fips` se desfragmente el disco). Si el sistema de ficheros es de tipo NTFS (WindosNT™, Windows2000™ o WindowsXP™), la cosa se complica, pues hasta la fecha no se conoce ninguna aplicación GPL que pueda hacer particiones de este tipo de sistema de ficheros. Así pues, frente a esta situación, de momento sólo se puede recurrir a aplicaciones comerciales (como puede ser PartitionMagic™).

ANOTACIONES

5.3.2. Arranque del sistema de instalación

Llegados a este punto, podemos empezar la instalación propiamente dicha. Para ello, arrancaremos el ordenador, nos aseguraremos de que el primer dispositivo a la hora de "bootar" sea la unidad de CD-ROM (entrando en la BIOS), y pondremos el CD1 en ella (dado que, en este caso concreto, vamos a usar el *flavor* **bf24**. También podríamos haber arrancado con el CD5 con los mismos resultados). Al cabo de unos momentos nos aparecerá una pantalla de bienvenida como la siguiente:

```
                    Welcome to Debian GNU/Linux 3.0!

This is a Debian CD-ROM. Keep it available once you have installed
your system, as you can boot from it to repair the system on your hard
disk if that ever becomes necessary (press <F3>for details).

For a "safe" installation with kernel 2.2.20, you can press <ENTER> to begin.
If you want additional features like modern hardware support, specify a
different boot flavor at the boot prompt (press <F3> to get an overview).
If you run into trouble or if you already have questions, press <F1>
for quick installation help.

WARNING: You should completely back up all of your hard disks before
   proceeding. The installation procedure can completely and cirreversibly
   erase them! If you haven't made backups yet, remove the CD-ROM
   from the drive and press <RESET> or <Control-Alt-Del> to get back
   to your old system.

Debian GNU/Linuxcomes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law. For copyright information, press <F10>.

Press <F1> for help, or <ENTER> to boot.

boot:
```

Podemos pulsar F1, sin demorarnos demasiado tiempo, ya que si no, la instalación empezará de forma automática y con el *flavour* **idepci**, *flavour* que se usa por defecto para acceder al índice de ayuda:

ANOTACIONES

```
HELP INDEX:

Press the function keys <F1> through <F10> for helpful information.

KEY   TOPIC

<F1>  This page, the help index.
<F2>  Prerequisites for installing this system.
<F3>  Boot methods for special ways of using this Rescue disk.
<F4>  Special boot arguments, overview.
<F5>  Special boot arguments for special machines.
<F6>  Special boot arguments for selected disk controllers.
<F7>  Special boot arguments for the install system.
<F8>  How to get help.
<F9>  Debian e-mail lists. <F10> Copyrights and warranties.

Press <ENTER> or type boot method, arguments, and <ENTER> to boot.
Press any of <F2> through <F10> for quick installation help.

boot:
```

Podemos pulsar cualquiera de las opciones que nos ofrece el menú, pero la más destacable es la F3, en la cual encontraremos los distintos *flavours* que admite el modo de arranque, y también cómo utilizar este mismo CD1 para usarlo como disco de rescate:

```
BOOT METHODS

The Rescue disk can be used for system recovery as well as for installation.
The rescue method will mount an existing root partition, while the other
methods will give you a small, standalone root system.

Available boot methods:
linux or idepci
   Start the installation -this is the default.
fbf24
   Start the installation with a Linux 2.4 kernel.
compact
   Start the installation with a kernel including PCI SCSI and IDE drivers.
vanilla
   Start the installation with a kernel including many drivers for older hardware
   (such as systems with ISA-based hardware).
rescue
   Boot and mount any root filesystem. The root filesystem must be given at
   the prompt, so e.g., type rescue root=/dev/hda1. (You can also use
   rescbf24, resccomp, or rescvanl.)

Press <ENTER> or type boot method, arguments, and <ENTER> to boot.
Press function key <F1> for the help index.

boot:
```

Tecleemos pues `bf24`, y empecemos con la instalación.

Una vez hecho esto, se cargará el *kernel* (veremos durante unos instantes las salidas por pantalla de las distintas pruebas que se hacen) e inmediatamente nos aparecerá una pantalla, *Choose The Language*, para que mediante los cursores seleccionemos el idioma de instalación.

A partir de ahora ya disponemos, pulsando la combinación de teclas ALT+F2 e INTRO, de un *shell,* que, aunque muy básico, puede usarse en cualquier momento de la instalación.

En la tty3 (ALT+F3) el *kernel* va dejando sus mensajes (es decir, el contenido del fichero `/var/log/messages`).

5.3.3. Configuración del idioma de instalación

Es muy recomendable seleccionar la opción `en Choose this and press Enter to proceed in English` para evitar posibles errores de traducción. En general, y si es posible, siempre es mejor trabajar con el idioma original, y así se hará durante el resto del taller; si bien, una vez más, se deja libertad al lector para que decida él mismo lo que le parece más adecuado; en este caso, que sea él quien elija el idioma que va a utilizar durante la instalación.

Escogemos el idioma inglés y pulsamos INTRO. En la pantalla *Choose Language Variant* siguiente se puede escoger qué inglés se quiere utilizar, nosotros usaremos el americano `English (United States)`.

Nota

El idioma de instalación no condiciona en modo alguno ni la configuración del teclado, el cual hasta el momento es `us`, ni el idioma de interacción con el sistema ni otras configuraciones. Esta opción sólo es válida durante el proceso de instalación.

5.3.4. Menú principal de instalación

Después de la pantalla de créditos *Release Notes*, y tras pulsar INTRO, entraremos en la interfaz *Debian GNU/Linux Installation Main Menu*, que nos servirá para configurar el resto de la instalación. Esta interfaz opera según unos criterios preestablecidos; así, siempre se nos sugerirá el próximo paso a dar para una instalación estándar, seguido, si procede, de alternativas a esta opción. No obstante, me-

ANOTACIONES

diante los cursores podemos desplazarnos y seleccionar cualquier opción de las que se nos ofrecen tras una línea en blanco (aquí se listan todas las opciones posibles de instalación, aunque algunas no estarán disponibles si en primer lugar no hemos utilizado otras; en principio, salvo casos excepcionales y por ser nuestra primera instalación, nunca usaremos estas opciones y dejaremos que la misma interfaz de instalación nos guíe por ellas mediante la sugerencia que nos haga en cada momento).

5.3.5. Configuración del teclado

Como primera sugerencia, la interfaz de instalación nos propone que configuremos el teclado `Configure the Keyboard`. Así lo hacemos. Pulsamos INTRO para acceder a la pantalla *Select a Keyboard*, y seleccionamos el teclado, `qwerty/es:Spanish;` pulsamos INTRO para que el dispositivo de entrada responda al mapeado de caracteres del teclado español. En la tty2 podemos comprobar que, efectivamente, ya disponemos de acentos, por ejemplo.

5.3.6. Partición del disco duro

Una vez seleccionado el teclado, el proceso de instalación nos devuelve al menú principal y nos propone que procedamos a fraccionar el disco duro, `Partition a Hard Disk`. Así lo hacemos. Pulsamos INTRO y entramos en la pantalla de selección de dispositivos que hay que fraccionar, *Select Disk Drive*. Si disponemos de más de un disco duro, mediante los cursores podremos seleccionar sobre cuál de ellos queremos operar. Una vez hecha la selección, pulsamos INTRO. Tras una pantalla de advertencia referente a las limitaciones de `Lilo`, *LILO Limitations*, y pulsando INTRO entramos en otra pantalla de consejo acerca de cómo se debe calcular el tamaño de las particiones si se tiene intención de utilizar el Reiser filesystem, *Note on additional space for Reiserfs Journal*. Pulsamos INTRO nuevamente y se arranca la aplicación `cfdisk` para crear y gestionar las particiones.

Los tamaños, sus características y el número de particiones dependen en gran medida del tipo de uso y de la cantidad de disco duro de que se disponga. Al tratarse de una instalación con fines educati-

ANOTACIONES

vos, se facilita seguidamente la información sobre las particiones que se crearán suponiendo que se trabaja sobre un espacio de entre los cinco y los quince gigabytes destinados a la nueva instalación.

Como mínimo hay que crear dos particiones: una primera para montar el sistema y otra de *swap* (es cierto que se puede hacer instalaciones sin *swap*, pero, tal como ya se ha dicho, no es en absoluto recomendable). Para aumentar la eficiencia del sistema, nosotros crearemos cinco particiones. La primera destinada a albergar la *root partition*; ésta no tiene que ser muy grande, por ello se le destinará alrededor de un diez por ciento del disco duro, preferentemente en una partición primaria; pero si no disponemos de ella, la podemos crear como partición lógica sin darle más importancia. La segunda se destinará a *swap*.

Se recomienda que ésta tenga un tamaño de entre 512kb y 1Mb; con estas dimensiones nos aseguraremos, salvo en casos excepcionales, de que nunca llegará a saturarse; esta partición también es preferible que sea primaria, pero si tiene que ser lógica, tampoco repercutirá en el rendimiento del sistema. La tercera partición estará destinada a albergar el directorio `/home/`, cuya finalidad es almacenar datos de usuario, y, dependiendo del tamaño del disco duro, se le puede asignar entre un diez y un veinte por ciento del disco duro, en función del número de usuarios y del uso que se va hacer del sistema. La cuarta partición será para el directorio `/usr/`; hay que tener presente que esta partición albergará gran parte del software que se instale; por lo que deberá tener un tamaño significativo, alrededor de un cuarenta por ciento del disco. El espacio restante se destinará al directorio `/var/`, donde se alojan librerías, ficheros de log, etc.

La distribución de particiones anterior es sólo una propuesta que tiene dos objetivos: por un lado, pretende mejorar el rendimiento que ofrece una instalación basada únicamente en dos particiones y, por otro lado, da más robustez al sistema. Entre otras ventajas, tener los datos repartidos entre distintas particiones hace que la corrupción de una de ellas no implique automáticamente la pérdida de toda la información del sistema. Obviamente, pueden crearse otras particiones u omitir algunas de las propuestas. (El orden de las particiones no afecta al comportamiento del sistema.)

ANOTACIONES

Por lo que se refiere a cfdisk, su funcionamiento se basa en el uso de los cursores, tanto para moverse por las particiones del disco (parte superior de la pantalla, mediante los cursores Up y down), como para seleccionar las posibles operaciones que se puedan realizar en cada momento (parte inferior de la pantalla, con left y right), ya que éstas irán variando según el estado de la partición seleccionada. Para crear una nueva partición, hay que seleccionar la opción [New]; a continuación, si aún se pueden crear particiones primarias (el número máximo es cuatro), se nos preguntará si la queremos crear como partición primaria o lógica; después debemos especificar, en Mb, el tamaño de la partición; y finalmente su ubicación física en el disco (es recomendable que antes de empezar a fraccionar el disco hagamos un pequeño esquema de cómo lo queremos hacer y crear las particiones a partir de éste, para poder así responder siempre [Beginning] a esta pregunta).

Una vez se han creado las particiones, hay que asignarles el tipo de sistema de ficheros. Por defecto las particiones se crean de tipo Linux, es decir, con código 0x83; en general, pues, esta opción por defecto ya nos será útil salvo para la partición destinada a *swap*. Para esta partición en concreto habrá que seleccionar la opción [Type], con lo que primero se nos mostrará, en hexadecimal, la relación entre códigos y *filesystem types*, y después se nos preguntará qué tipo de sistema de ficheros queremos asignarle a la partición seleccionada. Por defecto se nos propone la 0x82, es decir, Linux *swap*, que es la opción deseada. Si tenemos más de una instalación de GNU/Linux en el mismo ordenador, se puede utilizar la misma partición *swap* para todas ellas, ya que la información que se pueda almacenar en ella durante el funcionamiento del sistema es totalmente volátil. Bastará, pues, con pulsar INTRO y se nos retornará a la pantalla principal de cfdisk mostrándonos que, efectivamente, se ha realizado el cambio.

Una vez tengamos construida la tabla de particiones a nuestra medida, sólo habrá que seleccionar la opción [Write] para que ésta, tras una pregunta de confirmación final a la que responderemos yes, se escriba en la MBR y poder hacer efectivos los cambios. Ahora, pues, ya podemos abandonar cfdisk con la opción [Quit] y proseguir con la instalación.

ANOTACIONES

5.3.7. Inicialización y activación de la partición *swap*

Una vez hemos salido del `cfdisk`, retornamos al menú principal. Esta vez se nos propone que inicialicemos y activemos la partición de swap, `Initialize an Activate a Swap Partition`. Pulsemos INTRO para entrar en una pantalla, *Scan fo Bad Blocks?*, donde se nos pregunta si deseamos chequear la partición en búsqueda de posibles bloques defectuosos. Esta misma pantalla nos aparece cada vez que inicializamos y activamos una partición. Hay que advertir que el proceso de chequeo es lento, así pues, esta vez lo activaremos para ver su modo de proceder, pero para el resto de particiones dejaremos que cada uno tome la decisión que estime más acertada. La experiencia pone de manifiesto que para discos nuevos, o en buen estado, no es necesario ejecutar este proceso de prueba, puesto que los resultados siempre suelen ser positivos.

Una vez terminado el proceso de chequeo, entramos en una pantalla de confirmación final, *Are You Sure?*, antes de proceder a la inicialización y a la activación de la partición. Pulsemos INTRO para llevar a termino nuestro objetivo.

5.3.8. Inicialización y activación de una partición Linux

De vuelta al menú principal, éste nos sugiere que inicialicemos y activemos una partición de Linux, `Initialize and Active a Swap Partition`. Esta operación es indispensable, pues es en este punto, cuando inicialicemos, activaremos y montaremos la partición *root* (/). Pulsamos INTRO para entrar en la pantalla de selección del tipo de sistema de ficheros que deseamos para esta partición, *Choose Filesystem Type*. Nosotros decidimos que sea de tipo ext3. Una vez hecha esta selección, entramos en la pantalla, donde elegiremos la partición sobre la que queremos operar, *Select Partition*. Como se trata de montar la partición *root*, elegiremos la primera. Una vez hecho así, entraremos en la pantalla ya conocida de chequeo de bloques defectuosos, y a continuación en la pantalla de confirmación de operación. Tras la operación, el sistema de instalación nos pregunta si queremos montar en esta partición el *root* filesystem, *Mount as the Root Filesystem?*. Obviamente, pues así lo habíamos planeado, responderemos afirmativamente.

ANOTACIONES

5.3.9. Inicialización y activación de otras particiones Linux

Ahora el menú principal nos sugiere, como opción principal, que instalemos el *kernel* y sus módulos en la *root partition*, `Install Kernel and Driver Modules`. Nosotros aún no lo haremos, pues, tal como habíamos planeado, tenemos que inicializar, activar y montar el resto de particiones que hemos creado. Elegimos, pues, la primera alternativa que nos ofrece el menú principal `Initialize a Linux Partition`. Hecho esto, entraremos en la pantalla de selección de *filesystem*, ext3 nuevamente, y entramos en la pantalla de selección de partición. Optamos por la primera que se nos sugiere, y entramos en las pantallas de chequeo de bloques y de confirmación. Una vez terminada la operación, se nos pregunta qué partición queremos montar *Select Mount Partition*. Aquí la primera vez optaremos por la de `/home`, siguiendo el planteamiento inicial.

De nuevo en el menú principal, repetiremos las acciones anteriores para montar las particiones de `/usr` y de `/var`.

5.3.10. Instalación del *kernel*

Una vez terminado con el proceso de inicialización, activación y montaje de todas las particiones que habíamos creado, podemos proceder al próximo paso que el menú de instalación hace ya rato que nos sugiere, es decir, instalar el *kernel* y los módulos `Install kernel and Driver Modules`. Entramos ahora en la pantalla de selección del medio de instalación *Select Installation Medium*. Como estamos instalando el sistema utilizando el juego de CD, elegimos la primera opción. Se nos pide ahora el primer CD, *Please insert the CD-ROM*, pero como éste ya se halla en el lector, podemos continuar el proceso.

Entramos ahora en una pantalla en la que se nos pide dónde debe ir el sistema a buscar los ficheros del *kernel* para la instalación, pero como estamos realizando la instalación desde CD-ROM, sólo se nos da una opción. Así pues, pulsamos INTRO para instalar el *kernel* en el disco duro, en la *root partition*.

ANOTACIONES

5.3.11. Configuración de módulos

Llegados a este punto, el menú principal nos sugiere la configuración de los módulos del *kernel* `Configure Device Driver Modules`. Siguiendo este paso, entramos en una pantalla de advertencia sobre los módulos que ya lleva incorporado el *kernel* que se ha instalado Note about loaded drivers, el cual nos indica que si no encontramos ningún módulo que sea del todo necesario podemos proseguir con la instalación. En principio no debemos preocuparnos demasiado por los módulos, pues por ahora nuestro objetivo principal es instalar un sistema mínimo en nuestro PC, y luego ya lo iremos ampliando y mejorando. Así pues, podemos dar un paseo por el menú de módulos con el fin de adquirir conocimientos, pero, en principio, no activaremos ninguno (una vez instalado el sistema básico, podremos entrar en este mismo menú y activar cualquier módulo mediante el comando `modconf`).

5.3.12. Configuración del *hostname*

Es momento de dar un nombre al sistema que estamos instalando `Configure the hostname`. Por defecto se nos sugiere el nombre de `Debian`, *Choose the Hostname*. Podemos borrar esta sugerencia y dar un nombre que nos guste más compuesto de una sola palabra, `brau`, por ejemplo. (En este punto sólo hay que especificar el nombre local del sistema. Si éste va a formar parte de una red, y por tanto necesita de dominio, éste ya se especificará en el apartado de configuración de la red.)

5.3.13. Instalación del sistema base

Con toda esta información, el sistema ya puede hacer la instalación del sistema base; es lo que nos propone el menú principal, `Install the System Base`. Aceptamos pues esta propuesta. Nuevamente se nos pregunta por el medio de instalación, CD-ROM, la opción por defecto y seguidamente se nos requiere de nuevo el CD1 para proseguir; pulsamos, pues, INTRO y dejamos que el sistema lea los ficheros necesarios.

El siguiente paso es especificar el directorio desde donde se va a instalar el sistema base *Select Archive Path*. Sólo se nos muestra una op-

ANOTACIONES

ción, `instmnt`, que es el directorio virtual que se ha creado para tal fin (en realidad esta información se obtendrá del CD1, a partir de enlaces simbólicos de este directorio al directorio `/CD-ROM/`). Así pues, pulsemos INTRO y dejemos que finalmente empiece la instalación del sistema base en el disco duro.

5.3.14. Creación de un disco de arranque

Una vez terminado el proceso de instalación del sistema base, el menú principal nos sugiere hacer el sistema *bootable*, `Make System Bootable`. Sin embargo, como medida de seguridad nos decidimos por la primera alternativa que se nos ofrece, es decir, crear un disco de arranque. Así pues seleccionamos esta opción, `Make a Boot Floppy`, y pulsamos INTRO. Insertamos un disquete virgen en la disquetera y procedemos a crear dicho disco.

5.3.15. Instalación de Lilo

Una vez creado el disco de arranque, el menú principal nos sugiere que "rebootemos" el sistema. Pero también podemos hacer el sistema *bootable*, como se nos sugería anteriormente, es decir, dejar que escriba en la MBR la información necesaria para que se pueda arrancar la instalación que acabamos de realizar desde el propio disco duro. Para ello, debemos seleccionar la opción `Make System Bootable` y contestar afirmativamente, por lo general, a las propuestas que nos hace el sistema de instalación.

Realizar esta operación en este momento está plenamente justificada si en nuestro PC sólo hemos instalado GNU/Linux, o si creemos que el gestor de arranque Lilo será capaz de "bootar" otros posibles sistemas operativos que tengamos instalados (en el caso de tratarse de sistemas Microsoft^{TM}, puede instalarse Lilo, ya que la experiencia muestra que Lilo es perfectamente capaz de gestionar su arranque). Pero no debemos realizar esta operación, si ya disponemos de otro gestor de arranque (el cual habrá que configurar para que pueda arrancar el sistema que acabamos de instalar), o si queremos instalar otro gestor de arranque como puede ser *gups*, o si no estamos plenamente seguros de que Lilo será capaz de gestionar correctamente el arranque del resto de sistemas operativos que tengamos instalados.

ANOTACIONES

Si no hemos instalado Lilo, de momento podemos arrancar el sistema operativo mediante el disquete que hemos creado anteriormente, y en el próximo taller evaluaremos la posibilidad de instalar el gestor de arranque desde el propio sistema.

5.3.16. Reinicialización del sistema

Ha llegado el momento de reinicializar el sistema para arrancar el sistema base que hemos instalado en nuestro disco duro, y a partir de él empezar a personalizar nuestra primera instalación de GNU/Linux. Así pues, retiramos el CD1, o cuando se produzca la reinicialización, entramos en la BIOS para reestablecer la secuencia de dispositivos de arranque, preferiblemente disquetera, disco duro, CD-ROM, etc. para un sistema estándar), y el disquete si hemos instalado Lilo, y seleccionamos la opción `Reboot the System`. Tras una pantalla final de advertencias y recomendaciones que debemos leer atentamente, podemos proceder a la reinicialización. Si no se ha instalado Lilo, el disquete deberá permanecer en la disquetera, y debemos asegurarnos de que el PC arrancará desde él.

5.3.17. Arranque del sistema base

Si todo ha ido como es debido, tras reiniciar el sistema observaremos, tanto si hemos arrancado íntegramente desde el disco duro, como si se ha hecho inicialmente mediante disquete, que el sistema, al arrancar, ya usa información del disco duro. Transcurridos unos momentos, en los cuales irán apareciendo por pantalla los resultados de los distintos procesos que se ejecutan durante el arranque, el sistema entrará en una pantalla de bienvenida, y nos invitará a proseguir la instalación, recordándonos que podremos repetir este proceso en cualquier momento ejecutando el comando `base-config` como *root*. Pulsamos INTRO y proseguimos.

5.3.18. Configuración horaria

Inmediatamente después de la pantalla inicial de configuración, entramos en la de la configuración horaria del sistema, *lime Zone Configuration*. En ésta se nos mostrará la hora de la BIOS. Si esta hora

ANOTACIONES

coincide con la hora de nuestro huso horario, debemos contestar que no; por el contrario, si esta hora está establecida respecto al meridiano de Greenwich, debemos contestar que sí, para que el sistema haga los cálculos necesarios para situarnos en la hora de nuestro huso horario.

5.3.19. Configuración geográfica

Una vez establecido el criterio para el cálculo horario, debemos especificar, en las dos pantallas siguientes, *Time Zone Configuration*, nuestro emplazamiento geográfico. Si la hora de la BIOS está sintonizada según la hora local, y así lo hemos especificado en el apartado anterior, las respuestas a estas preguntas no tendrán ninguna relevancia. Sí que es importante contestar correctamente a estas preguntas si el horario de la BIOS se ha establecido según el criterio GTM. (Siempre es posible reconfigurar estos parámetros mediante el comando `tzconfig`).

5.3.20. Establecimiento de la política de *passwords*

Una vez terminados los procedimientos para la configuración horaria del sistema, ha llegado el turno de establecer la política de *passwords*, *Password Setup*. Ésta se compone de posible habilitación de los sistemas md5 y shadow, y del establecimiento del *password* de *root*, y de la creación de una cuenta de usuario.

Sistema md5

En primer lugar el sistema nos pregunta si queremos instalar el sistema de *passwords* md5. Si no tenemos sospecha de problemas de incompatibilidades (NIS, etc.), podemos contestar que sí, ya que es buena idea poder usar *passwords* con más de ocho caracteres.

Sistema shadow

Una vez más, si no tenemos sospecha de posibles incompatibilidades (este sistema no suele implicar problemas de incompatibilidades), es bueno habilitar esta opción, ya que así nos aseguramos de que el fichero que contiene los *passwords* encriptados (`/etc/shadow`) sólo

ANOTACIONES

será accesible por el *root*. Esto supone una medida más de protección para nuestro sistema.

Password de *root*

Ahora debemos entrar el *password* de *root*. Es importante seguir las recomendaciones para seleccionar este *password* y acordarse de él. Con el objetivo de confirmar que se ha entrado el *password* deseado y no ha habido posibles errores de tecleo, se nos pide que lo volvamos a escribir a modo de confirmación.

Nota

Frente al olvido del *password* de *root*, hay distintas estrategias a seguir. Una de ellas puede ser utilizar el CD-ROM de instalación, inicializar una instalación y, justo después de haber configurado el teclado, acceder a la tty2; una vez allí, podemos montar la *root partition* de nuestro disco duro y mediante un editor eliminar el contenido del campo de *password* encriptado del fichero `/etc/passwd`. Cuando volvamos a reiniciar el sistema, el superusuario no tendrá *password*, por esta razón, lo primero que debemos hacer es asignarle inmediatamente uno.

Creación de una cuenta de usuario

Tal como ya se ha remarcado, trabajar siempre como *root* es una mala política por distintos motivos. A consecuencia de ello, el sistema nos recomienda crear una cuenta de usuario normal. Se nos pedirá el nombre de esta cuenta, si queremos añadir datos suplementarios de la misma (por defecto Debian user), y su *password*, que, por la misma razón, debemos escribirlo dos veces.

5.3.21. Últimas configuraciones

Paquetes PCMCIA

Una vez creado el usuario normal, si no disponemos de dispositivos PCMCIA (típicos en ordenadores portátiles), el sistema lo detectará automáticamente y nos sugerirá que confirmemos que los paquetes asociados a estos dispositivos pueden ser eliminados.

Configuración de un sistema PPP

Si disponemos de un módem, puede ser buena idea que procedamos a su configuración, aunque también se puede hacer una vez estemos en el sistema mediante la aplicación `pppconfig` ejecutada como *root*. Obviamente, si no disponemos de este dispositivo, responderemos negativamente a esta pregunta. Si no requerimos de los servicios de PPP, podemos desinstalar sus paquetes: `ppp`, `pppconfig`, `pppoe` y `pppoeconf`.

ANOTACIONES

5.3.22. Configuración de apt

En esta sección se nos pregunta acerca de las fuentes donde apt deberá ir a buscar la información para construir su base de datos sobre los paquetes (dependencias, estado de instalación, etc.). Al estar haciendo una instalación íntegramente basada en CD-ROM, escogemos la primera opción, y, tras poner el CD1 en el lector, aceptamos la ruta que nos sugiere para acceder a él. Transcurridos unos momentos en los cuales apt extrae la información relativa a los paquetes que contiene este CD, se nos preguntará si queremos añadir el contenido de otro CD a la base de datos. Una vez hayamos introducido el CD2 en el lector, contestamos `yes` a esta pregunta. Repetimos esta operación hasta que apt haya procesado el contenido del séptimo CD-ROM. Las operaciones sobre la gestión de contenidos de los CD-ROM se pueden hacer también desde el sistema mediante la aplicación `aptCD-ROM`.

Una vez finalizado el escaneo de los contenidos de los siete CD-ROM, se nos preguntará si queremos añadir alguna otra fuente de donde apt pueda obtener paquetes. De momento contestaremos que no, al igual que a la siguiente pregunta respecto al chequeo de seguridad de los contenidos de los paquetes (no es necesario de momento, ya que, al estar usando únicamente paquetes de la distribución oficial, en principio sus contenidos ya son seguros).

5.3.23. tasksel y dselect

A continuación se nos pregunta si queremos que se ejecuten los programas de selección de paquetes, primero `tasksel` y luego `dselect`. Responderemos que no, pues estos programas también los podremos ejecutar desde la línea de comandos una vez hayamos terminado la instalación.

Aquí termina la instalación básica, como último punto, y ya respondiendo a preguntas formuladas por `apt`, se nos informará de los paquetes que se van a instalar y los que se van a desinstalar. Una vez aceptado este proceso, se nos preguntará si queremos borrar los paquetes `.deb` de nuestro disco duro. (Estos archivos se almacenan en /var/cache/). Al disponer de ellos en CD-ROM, podemos contestar que sí, ya que su presencia sólo implica la ocupación de espacio en el disco duro.

ANOTACIONES

Automáticamente, se arranca el programa de configuración de `exim`. Al no ser objeto de este taller la configuración de este programa de gestión de correo, escogemos la opción `5` para salir del programa. Finalmente, tras unos momentos nos aparecerá por pantalla:

```
Debian GNU/Linux 3.0 brau tty1
brau login:
```

Así, tenemos instalado un sistema básico estándar en nuestro ordenador. Ahora empieza la tarea de construir un sistema a nuestra medida. Es muy importante, llegados a este punto, distinguir entre dos tipos de sistemas: los de desarrollo y los de producción.

Un sistema de desarrollo es aquel que está destinado a experimentación, donde se hacen pruebas y experimentos, y donde no priman ni la estabilidad ni la eficiencia, es, en definitiva, un sistema destinado a la adquisición de conocimientos sobre el propio sistema. Contrariamente, en un sistema destinado a producción la eficiencia y la estabilidad son las características que más peso tienen. Así pues, debemos asegurarnos de que contenga única y exclusivamente aquellos paquetes estrictamente necesarios, ya que el hecho de tener paquetes innecesarios instalados va en detrimento de la eficiencia. La estrategia que habrá que seguir antes de montar un sistema de producción (un servidor de web, un servidor de aplicaciones, etc.) pasa siempre por trabajar primero sobre un sistema de desarrollo, donde se ensayarán y probarán distintas tácticas para poder sacar conclusiones de cara a montar el sistema de producción.

Nuestro primer sistema será, evidentemente, un sistema de desarrollo, tanto porque no tenemos en mente que sirva para cubrir ninguna necesidad en concreto, como porque es nuestra primera experiencia de montaje de un sistema, cuya finalidad es, únicamente, la obtención de conocimientos. Instalaremos y desinstalaremos distintos paquetes, haremos pruebas con distintas configuraciones, etc. y esto, claro está, perjudica directamente la eficiencia y la estabilidad del sistema. Así pues, animamos al lector, a que, una vez finalizado el módulo y haya obtenido ya una idea global de lo que es instalar un sistema, con todo lo que ello implica, reinstale de nuevo todo el sistema partiendo de cero para adaptarlo estrictamente a sus necesidades.

ANOTACIONES

5.4. Instalación de Debian Woody por red

Muchos de los pasos para realizar una instalación por red son comunes a los de una instalación mediante CD-ROM, así pues, destacaremos y nos centraremos en aquellos aspectos que difieren del proceso anteriormente descrito.

5.4.1. Particularidades de una instalación por red

La diferencia básica entre una instalación hecha mediante un juego de CD-ROM y una por red radica en la ubicación de los paquetes a la hora de instalarlos. Mientras que en una instalación hecha a partir de CD-ROM, habrá que insertar en el lector al menos uno de ellos cada vez que se instale un paquete nuevo para poder extraer de él los datos necesarios. En una instalación por red los paquetes se obtienen remotamente, hecho que nos permite, en primer lugar, acceder a la última versión de éstos, y poder actualizar todos los que tengamos instalados con una simple línea de comandos. (`apt-get upgrade`)

Por otro lado, para realizar una instalación por red, generalmente basta con un juego de disquetes, o con un único CD-ROM, que contenga la información necesaria para poder arrancar un sistema operativo básico sobre el cual se hará correr el programa de instalación (que ya está incluido en el juego de disquetes o en el CD-ROM, junto con los módulos que nos puedan ser necesarios para configurar el acceso a la red), y a partir de aquí, el resto de información se obtendrá remotamente.

5.4.2. Aspectos comunes de los distintos métodos de instalación

Como ya se ha dicho, muchos de los pasos a seguir para hacer una instalación por red son comunes a los datos para hacer una instalación mediante el juego de CD-ROM de la distribución. Así pues, entramos en el programa de instalación del mismo modo en que lo hemos hecho anteriormente ([5.3.1.] y [5.3.2.]) e inicialmente seguimos los mismos pasos para configurar el idioma de instalación, el teclado, fraccionar el disco duro y activar las particiones ([5.3.3.], [5.3.5.], [5.3.6.], [5.3.7.], [5.3.8.] y [5.3.9.]), y para instalar el *kernel* ([5.3.10.]). En el apartado `Select Installation Médium` optamos por la opción `/dev/`

ANOTACIONES

`fd0`, e inmediatamente se nos pedirá que insertemos el `rescue disk` seguido de los discos de *drivers*. Las imágenes de estos discos se hallan en `/CD-ROM/dists/woody/main/disks-i386/3.0.23-2002-05-21/images-1.44/ bf2.4.`. Llegados a este punto, tendremos que comprobar si el *kernel* ha reconocido nuestra tarjeta de red y, si no es el caso, procedemos a instalar y configurar el módulo pertinente para hacerla operativa. En muchos casos la configuración, paso de parámetros, se puede hacer de modo automático mediante `modprobe`, programa que se puede lanzar desde la misma interfaz de instalación después de la selección de un módulo. Una vez hecho esto, debemos configurar la red, especificar de qué sitio obtendremos las fuentes, y a partir de aquí seguiremos los mismos pasos que con el CD-ROM para terminar su instalación.

5.4.3. Instalación del módulo de red

Éste es un punto clave para poder realizar la instalación por red, ya que es aquí donde, si el *driver* de nuestra tarjeta de red no ha sido compilado dentro del *kernel*, debemos seleccionar el módulo necesario para tener acceso a ella. En primer lugar, pues, debemos averiguar si nuestra tarjeta de red ya ha sido detectada durante el proceso de arranque y se ha cargado su correspondiente *driver*. Para hacerlo, accedemos al segundo terminal (ALT+F2) y ejecutamos el comando `dmesg`. Ahora debemos buscar, entre las muchas líneas que nos ha devuelto este comando, si hay algunas que hacen referencia a nuestra tarjeta de red. A modo de ejemplo, para una tarjeta RTL-8029 (Realtek Semiconductors) se obtiene:

```
brau:˜# dmesg
.
.
.
ne2k-pci.c:v1.02 10/19/2000 D. Becker/P. Gortmaker
  http://www.scyld.com/network/ne2k-pci.html
PCI: Found IRQ 11 for device 00:0b.0
PCI: Sharing IRQ 11 with 00:07.2
eth0: RealTek RTL-8029 found at 0xe400, IRQ 11, 52:54:00:DB:FB:D4.
.
.
.
```

Si la búsqueda ha resultado infructuosa, en primer lugar debemos determinar qué tarjeta de red tenemos. Para ello, lo mejor es acudir a la documentación incluida con ella o a su inspección visual. Si esto no es posible, hay otras estrategias para determinar cuál es nuestra tarjeta, como puede ser, pulsar ALT+F2 para acceder a la consola e investigar el contenido del fichero `/proc/pci` (mediante `cat`, por ejemplo), o podemos recurrir a la información que nos puede proporcionar algún otro sistema operativo que tengamos instalado en el ordenador.

Una vez conozcamos qué tipo de tarjeta de red es la que tenemos, debemos averiguar qué módulo es el que nos servirá para acceder a la tarjeta. La estrategia más segura para este fin es recurrir a cualquier buscador, por ejemplo http://www.google.es, entrar palabras clave sobre nuestra tarjeta (*referencia de la tarjeta* `NIC linux module`, por ejemplo) y leer algunas de las páginas encontradas. También se puede recurrir a las páginas de las principales distribuciones de linux, y entrar la referencia de la tarjeta en sus buscadores. Como último recurso, se puede recurrir a la documentación de módulos de red del *kernel*, donde se especifica, para todas las tarjetas soportadas, el módulo correspondiente. También es bueno saber si el fabricante ha desarrollado su propio módulo. Llegar a encontrar un módulo para una tarjeta puede ser una tarea muy complicada, incluso imposible, ya que puede suceder que no haya soporte para dicha tarjeta o que haya que recurrir a métodos avanzados para poderlo configurar; por este motivo, se recomienda utilizar siempre tarjetas lo más estándar posible.

Una vez hayamos averiguado el módulo que necesitamos, después de instalar el *kernel* ([5.3.10.]), debemos seleccionar la propuesta que nos sugiere el menú principal `Configure Device Driver Modules`. Tras una pantalla de advertencia, donde se nos recuerda que muchos *drivers* ya están incluidos en el *kernel*, entraremos en la pantalla de selección de módulos *Select Category*, (podemos acceder a esta interfaz en cualquier momento, ejecutando el comando `modconf`. Este comando sirve como *fornt-end* para la administración de *drivers* que han sido compilados de forma modular junto al *kernel*) y mediante los cursores seleccionaremos la opción `kernel/drivers/net`. Una vez dentro de la pantalla de selección de módulos de tarjeta de red Select Kernel/drivers/net modules, seleccionamos otra vez con los cursores el módulo que necesitamos. Tras responder que sí a la pregunta sobre si realmente queremos instalar

ANOTACIONES

dicho módulo, podemos dejar que *el autoprobe* configure el módulo por nosotros, si no hay que pasar ningún parámetro en concreto al módulo en cuestión. Pasados unos instantes, recibiremos el mensaje indicándonos si el módulo se ha instalado correctamente o no.

5.4.4. Configuración de la red

Una vez sabemos que la tarjeta de red está operativa, en el menú principal de la instalación seguiremos el paso sugerido `Configure the Network` para proceder a la configuración de la red. En primer lugar deberemos configurar el nombre del *host* (antes se ha sugerido `brau`, por ejemplo), sin entrar el dominio. A continuación, deberemos entrar la IP, la máscara de red, el *gateway*, el nombre de dominio, y los servidores de DNS, hasta un número de tres, separados por espacios.

5.4.5. Configuración de apt

Una vez se ha configurado la red, los siguientes pasos que hay que seguir son idénticos a los seguidos en la instalación por CD-ROM, hasta llegar a la configuración de apt *Apt Configuration*. Llegados a este punto, en vez de elegir la opción de `CD-ROM`, optaremos por la opción de red que más nos convenga. A efectos prácticos es lo mismo seleccionar el protocolo http que el ftp. Después de realizar la selección se nos preguntará si deseamos usar paquetes de tipo **non-US**; en principio, y salvo problemas legales, responderemos que sí. Respecto a la respuesta sobre la pregunta siguiente, referente al uso de paquetes ***nonfree***, ya se deja que el mismo lector tome la decisión según sus principios éticos. Acto seguido se nos pregunta de qué estado debe ser el *mirror* del cual apt deberá obtener los paquetes, debemos escoger siempre el que nos sea accesible de forma más rápida, que acostumbra a ser el más próximo a nosotros geográficamente hablando. Una vez seleccionado el estado, se nos pide que seleccionemos un servidor en concreto (la aplicación `netselect` destinada a facilitar la elección de servidores de paquetes según el criterio de velocidad de acceso). Cuando se ha resulto esta cuestión, se nos muestra la pantalla de configuración de acceso a través de proxy; si no tenemos que usar este servicio, dejaremos en blanco la línea.

ANOTACIONES

Es importante que los paquetes críticos, en cuanto a seguridad se refiere, se obtengan de servidores seguros. Por esta razón, se recomienda que éstos se obtengan en concreto de `security.debian.org`.

Terminados todos estos pasos, apt conectará con el *mirror* que le hemos especificado para configurar su base de datos. A partir de este punto, para el resto de la instalación seguiremos los mismos pasos que en la instalación realizada mediante CD-ROM.

5.5. Conclusión

En este taller hemos aprendido a instalar GNU/Linux en nuestro ordenador. Aunque por ahora nuestro sistema sea muy básico, el objetivo del taller se ha cumplido plenamente, ya que hemos sentado las bases para poder empezar a sacar partido de la flexibilidad y potencia de este sistema operativo. En el próximo taller aprenderemos cómo configurar el sistema e instalar nuevas aplicaciones para ir adaptándolo y dotándolo de todas las herramientas que estimemos necesarias para cubrir nuestras necesidades.

ANOTACIONES

6. Configuraciones básicas

6.1. El sistema de *login*

Si todavía no tenemos configurado el entorno gráfico, cuando arrancamos un sistema GNU/Linux nos aparece una pantalla de *login* donde se pide al usuario que se identifique antes de empezar a utilizar el sistema. De hecho, la mayoría de distribuciones lanzan varias consolas a las que podemos acceder a partir de ALT+F1, ALT+F2, etc. Ello nos permite trabajar simultáneamente con diferentes cuentas a la vez, tener varias sesiones abiertas para ejecutar diferentes programas, etc.

El programa que se encarga de gestionar cada una de estas consolas es el `getty`. Lo único que hace este programa es abrir una conexión con el dispositivo adecuado (en el caso de las consolas de la pantalla, es el `/dev/ttyX`, donde la "`X`" es el número de consola) y lanzar la aplicación de *login*. Este mecanismo nos permite mucha flexibilidad, ya que el mismo programa `getty` permite comunicarse con diferentes dispositivos, de forma que podríamos conectar un terminal por el puerto serie del ordenador, montar una consola utilizando la línea telefónica y un módem, etc.

Antes de lanzar la aplicación de *login*, se muestra un mensaje de bienvenida por pantalla. Este mensaje lo podemos configurar en el fichero `/etc/issue`, escribiendo lo que queramos. En este mismo fichero también podemos mostrar algunas de las variables del sistema referenciándolas como:

"`\d`"	la fecha actual
"`\s`"	el nombre del sistema operativo
"`\l`"	el número de consola
"`\m`"	la arquitectura del ordenador
"`\n`"	el nombre del ordenador
"`\o`"	el nombre del dominio
"`\r`"	la versión del sistema operativo
"`\t`"	la hora actual
"`\u`"	número de usuarios activos en el sistema

ANOTACIONES

Una vez entramos en el sistema, el programa de `login` se encarga de mostrarnos el mensaje del día. Este mensaje es lo que hay escrito en el fichero `/etc/motd`, que también podemos cambiar. Este mecanismo es muy útil para informar a todos los usuarios de algún evento determinado, avisarles de algún problema, etc. Si un usuario quiere suprimir este mensaje, puede hacerlo creando un fichero vacío llamado `.hushlogin` en su directorio *home*. Después de mostrar este mensaje, el proceso de *login* lanza el *shell* configurado por defecto para el usuario. Lo primero que hace el intérprete de comandos es ejecutar el contenido del fichero `.profile` (que debe estar en el directorio *home* del usuario). Este fichero sirve para que se ejecuten las instrucciones configuradas siempre que el usuario entre en el sistema. Además de este `~/.profile`, también tenemos el `/etc/profile`, que se ejecuta para todos los usuarios del sistema y resulta muy útil para poder configurar a todos las opciones que deseamos sin tener que poner las instrucciones dentro de cada uno de los `.profile` de los usuarios.

6.2. Explorando el bash

Si bien el fichero `.profile` es ejecutado, sea cual sea el *shell* que utilicemos, los archivos `.bashrc` o `.bash profile` se suelen ejecutar sólo cuando utilizamos el bash (aunque se puede configurar a partir del mismo `.profile` del usuario, que es donde se llama a la ejecución de este archivo). Vamos a ver algunas de las instrucciones que podemos encontrar en estos ficheros:

```
#CONFIGURACIONES BÁSICAS
mesg n
umask 022
#PATH
export PATH= /usr/local/sbin:/usr/local/bin:/usr/
             sbin:/usr/bin:/sbin:/bin:/usr/bin/X11
#PROMPT
export PS1='\h:\w\$ '
#ALIAS DE USUARIO
alias l='ls --color=auto'
alias ll='ls --color=auto -al'
alias rm='rm -i'
alias cp='cp -i'
```

ANOTACIONES

```
alias mv='mv -i'
alias v='vim'
```

Como vemos, en este archivo podemos definir lo que queramos. Las dos primeras instrucciones del fichero anulan la entrada de mensajes de otros usuarios y configuran los permisos que tendrán los nuevos ficheros que vamos a crear. La siguiente instrucción es la definición del **PATH**. El PATH son los directorios donde tenemos los comandos, programas, aplicaciones, etc. que queremos poder llamar desde cualquier sitio de la jerarquía de sistema de ficheros sin necesidad de escribir su ruta completa (cada directorio del PATH lo separamos con "`:`"). La siguiente declaración es la del ***prompt*** del sistema. El *prompt* es la línea que aparece en el *shell* antes del carácter "`#`" (para el *root*) o "`$`" (para los otros usuarios). Podemos configurarnos este *prompt* de la forma que queramos utilizando las siguientes variables del sistema:

"`\d`"	la fecha del sistema
"`\h`"	el nombre de la máquina
"`\s`"	el *shell* que utilizamos
"`\u`"	el nombre del usuario
"`\v`"	la versión del bash
"`\w`"	el directorio actual
"`\!`"	el número de historia del comando
"`\$`"	aparece "`#`" si somos el *root* o "`$`" para los otros usuarios

Finalmente, tenemos los **alias** de usuario. Los alias son sinónimos, generalmente para los comandos que más utilizamos (para no tener que escribirlos completamente). Por ejemplo, en uno de los alias que teníamos en el ejemplo definíamos que al escribir "`l`" se ejecutara "`ls --color=auto`". De esta forma, podemos utilizar largos comandos sin tener que estar escribiéndolo todo cada vez que los utilizamos.

Tanto en la definición del PATH como en la del *prompt* hemos utilizado el comando `export`. Este comando nos permite definir lo que llamamos una **variable de entorno**. Estas variables son utilizadas por el *shell* para realizar ciertas operaciones, guardar algún tipo de información, etc. Podemos ver todas las que hay declaradas con el mismo comando `export`. Con `set` y `unset` también podemos

Nota

Si queremos ejecutar los programas del directorio desde donde estamos situados sin necesidad de poner "`./`" al principio, podríamos añadir esta entrada en la declaración del PATH. Igualmente, si en el PATH no hay el programa que necesitamos ejecutar, podemos especificar la ruta completa del mismo en la línea de comandos. De todas formas, no es recomendable añadir "./" al PATH porque puede representar un agujero de seguridad.

Nota

Podemos ver todos los alias definidos a partir del mismo comando `alias`.

Nota

Con "`echo $NombreVariable`" podemos ver el contenido de estas variables y atributos.

manipular otros atributos que tiene el intérprete de comandos. Algunas de estas variables y atributos, que tiene por defecto el bash, son:

- PWD: directorio actual.
- BASH_VERSION: versión del bash que utilizamos.
- RANDOM: genera un número aleatorio diferente cada vez que mostramos su contenido.
- SECONDS: número de segundos que han pasado desde que hemos abierto el *shell*.
- *HOSTNAME*: nombre del sistema.
- OSTYPE: tipo de sistema operativo que estamos utilizando.
- MACHTYPE: arquitectura del ordenador.
- HOME: directorio *home* del usuario.
- HISTFILESIZE: tamaño del fichero de historia (número de comandos que se guardan).
- HISTCMD: número de comando actual en la historia.
- HISTFILE: fichero donde se guarda la historia de comandos (generalmente .bash_history del directorio *home* del usuario).

Con la manipulación de estas variables podemos personalizar mucho más nuestro intérprete de comandos para adaptarlo a nuestros gustos y necesidades.

6.3. El sistema de arranque

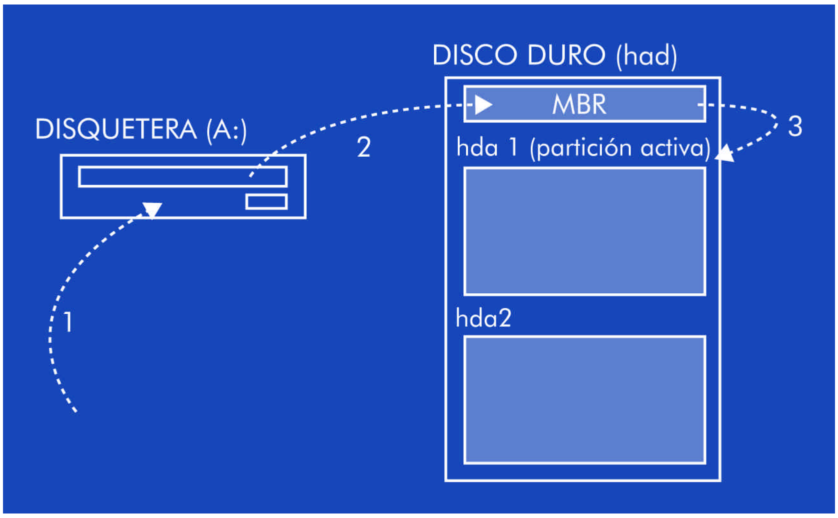
Aunque con la instalación del sistema operativo ya se lleva a cabo la configuración e instalación de un sistema de arranque, en esta sección veremos con detalle qué opciones nos proporcionan y cómo de-

ANOTACIONES

bemos personalizarlos y adaptarlos a nuestras necesidades. Aunque existen muchos, el Lilo y el Grub son los más utilizados en los entornos GNU/Linux, con lo cual sólo nos centraremos en ellos.

Antes de entrar en detalles sobre la configuración de estos dos programas, explicaremos cómo funciona el sistema de arranque de un PC estándar. Tal como ya sabemos, a partir de la BIOS o EFI del ordenador podemos configurar la secuencia de arranque del mismo. En general, esta secuencia suele empezar buscando en la disquetera y sigue con el CD/DVD y el disco duro. Aunque podemos instalar el Lilo o el Grub en un disquete o en el sector de arranque de un CD, es más usual instalarlo en el disco duro para no tener que introducir el disco cada vez que arrancamos nuestro ordenador.

Cuando el sistema de arranque del ordenador va a buscar en el disco duro, lo primero que inspecciona es si la MBR (*Master Boot Record*) del primer disco duro (máster del primer canal IDE o el primer disco del canal SCSI) contiene alguna indicación del sistema que hay que cargar. La MBR es la primera pista del disco duro, allí donde se guarda la información de las particiones configuradas y, opcionalmente, el programa encargado de iniciar el sistema operativo. Si aquí no se encuentra este programa, se inspecciona el sector de arranque de la partición activa del disco. Siempre que queremos instalar un programa en el sistema de arranque del ordenador debemos situarlo en alguna de estas zonas. En la siguiente figura podemos ver todo este proceso cuando en la secuencia de arranque primero es la disquetera y después el disco:



ANOTACIONES

Siempre que instalamos un sistema de arranque, debemos tener en cuenta que el orden con que se realiza esta secuencia es importante: si instalamos uno en la MBR y otro en la partición activa, se ejecutará el de la MBR porque la BIOS o EFI inspecciona primero esta zona. Si no tenemos partición activa, debemos situar el programa de arranque en la MBR. De todas formas, lo más recomendable es instalar siempre el programa en la MBR porque es lo primero que se inspecciona. Aunque podamos tener instalados otros sistemas operativos en otros discos, debemos instalar el Lilo o Grub en alguna de estas zonas. En la configuración del programa ya le indicaremos dónde están situados los operativos que queremos cargar.

6.3.1. Lilo

El Lilo nos permite múltiples configuraciones diferentes para realizar casi cualquier acción con el sistema de arranque del ordenador. Para no extendernos innecesariamente, en esta sección sólo explicaremos las opciones más importantes para poder configurar un sistema de arranque de múltiples operativos teniendo en cuenta la seguridad del mismo. En el manual del programa y de su fichero de configuración (“`man lilo`” y “`man lilo.conf`”) encontraremos la especificación completa para todas las otras opciones. El fichero de configuración del Lilo es el `/etc/lilo.conf`. Cuando ejecutamos `lilo`, el programa busca toda la información en este fichero e instala el programa de arranque en la zona que le indicamos. En este fichero de configuración tenemos dos secciones: la global y las locales a cada sistema operativo. En la sección global encontramos todas las configuraciones generales para el programa. En cada línea podemos poner una directiva de configuración para indicar algún tipo de información al programa. Las directivas más importantes de esta sección son las siguientes:

- `boot = DISPOSITIVO`: le indicamos en qué dispositivo queremos instalar el Lilo.

Si queremos instalarlo en la MBR del primer disco del canal IDE, en `DISPOSITIVO` pondremos `/dev/hda`, si lo queremos en la primera partición, `/dev/hda1`, etc. También tenemos la opción de instalarlo en un disquete, indicando el dispositivo `/dev/ fd0`.

ANOTACIONES

- `default = NOMBRE`: indicamos el sistema operativo que se cargará por defecto. El NOMBRE debe corresponder a alguna de las etiquetas configuradas en cada sistema de las secciones locales.

- *`prompt`*: con esta directiva indicamos al programa que antes de cargar el sistema muestre un *prompt* por pantalla para que podamos seleccionar el que queramos. Si no activamos esta opción, no podremos seleccionar el operativo que hay que cargar y automáticamente se iniciará el que tengamos por defecto.

- `delay = TSECS`: especifica el número de décimas de segundo que el Lilo esperará antes de cargar el sistema operativo por defecto (la directiva `default` debe existir). Esta opción es válida cuando sólo tenemos un sistema operativo en el disco; si utilizamos *`prompt`*, tendremos que configurarlo con `timeout`.

- `timeout = TSECS`: décimas de segundo que el Lilo esperará en el *prompt* antes de arrancar el sistema operativo por defecto (deben existir la directiva de *`prompt`* y `default`).

- `lba32`: esta opción nos sirve para poder arrancar cualquier partición del disco duro.

Si desactivamos esta directiva, no podremos arrancar sistemas operativos en particiones situadas en sectores superiores del disco (a partir del cilindro 1024) y deberemos poner el arranque del sistema en una pequeña partición al inicio del disco (más información en "`man lilo.conf`"). Todas las BIOS a partir de 1998 soportan esta opción, de forma que es muy recomendable utilizarla para ahorrarnos problemas.

Las directivas locales son las que corresponden con cada uno de los sistemas operativos instalados en las particiones de nuestros discos. La primera directiva debe ser o bien `image` u `other`. Las explicamos a continuación (junto con otras directivas posibles):

- `image = CAMINO`: el `CAMINO` nos indica el archivo que contiene la imagen núcleo del sistema operativo. Esta opción sólo es válida para sistemas GNU/Linux o similares. Generalmente, las distribuciones crean un directorio `/boot/` donde se pone la imagen del

Contenido complementario

El sistema de arranque de un sistema GNU/Linux o similar nos permite, con un mismo sistema de ficheros, cargar varios núcleos diferentes. De este modo, sin la necesidad de instalar de nuevo el sistema, podemos trabajar con diferentes núcleos. Para configurarlo sólo tendríamos que especificar dos secciones locales poniendo, en cada una, qué núcleo utilizaremos.

núcleo del sistema, aunque también es usual poner un enlace en la raíz llamado `/vmlinuz` (o similar). En esta directiva podemos especificar cualquiera de los dos.

- `other = CAMINO`: dispositivo que contiene un sistema operativo diferente a GNU/Linux o similar.
- `label = NOMBRE`: etiqueta para el sistema operativo. Esta directiva es obligatoria para todas las secciones locales que tengamos configuradas, ya que si no, no podríamos identificarlas de ninguna forma.
- `alias = NOMBRE`: sinónimo para la misma sección local. Podemos tener tantos como queramos.
- `root = DISPOSITIVO`: esta opción sólo es válida para los sistemas GNU/Linux o similares. Se le especifica en qué dispositivo está situada la raíz del sistema de ficheros (root filesystem). Esta opción se pasa como argumento al cargar el núcleo del sistema y es opcional; si no se le pasa nada, se coge la que tiene configurada por defecto la imagen del núcleo.

Hay algunas de las directivas del Lilo que también pueden ser configuradas directamente sobre la imagen del núcleo del operativo. Con el comando `rdev` podemos configurar las posibles opciones de una imagen núcleo Linux. Algunas de ellas son la raíz del sistema de ficheros (si no lo configuramos con el Lilo, se coge el que hay en la imagen), el modo vga, etc.

Si queremos proteger adecuadamente el sistema de arranque del ordenador, debemos añadir algunas directivas más en este fichero de configuración. Lo primero que debemos hacer es proteger la BIOS o EFI con una contraseña y configurar la secuencia de arranque para que sólo se pueda realizar a partir del disco duro. Con esto pasamos toda la responsabilidad al Lilo. A partir de las directivas `password = CONTRASEÑA` y `restricted`, podremos configurar la seguridad del mismo de cinco maneras diferentes:

Contraseña global	Debemos poner la directiva de `password` en la sección global y al arrancar cualquiera de los sistemas se pedirá la contraseña.

ANOTACIONES

Contenido complementario

Con el Lilo o el Grub podemos pasar parámetros al núcleo Linux en el momento de arrancar. Esto es muy útil cuando queremos realizar alguna operación específica en el sistema; por ejemplo, pasando `single` o `1` se iniciaría el sistema en el *runlevel* 1, con `root=/dev/hda3` especificaríamos la raíz del sistema de ficheros, etc.

Contraseñas locales	En los sistemas que queramos que nos pida contraseña debemos poner la directiva *password* y sólo al arrancar éstos se pedirá la misma.
Contraseña restringido global	Debemos poner la directiva de `password` y `restricted` en la sección global y sólo al arrancar algún sistema pasando algún parámetro al núcleo se pedirá la contraseña.
Contraseña global y restringido local	La contraseña se pone en la sección de global y sólo ponemos el `restricted` en los sistemas que queremos que se pida contraseña al pasar algún parámetro al núcleo.
Contraseña y restringido local	Es lo mismo que la configuración anterior, pero con la ventaja de que podemos configurar diferentes contraseñas para los diferentes sistemas que queremos arrancar.

Debemos tener en cuenta que si utilizamos la directiva `password`, la contraseña se escribe en el fichero como un texto, de forma que deberemos eliminar los permisos de lectura del `lilo.conf` para todos los usuarios menos el *root*. Finalmente, después de escribir este fichero de configuración podemos instalar el Lilo en el sector de arranque configurado. Para hacerlo, sólo debemos ejecutar “`lilo`”. Si quisiéramos desinstalarlo deberíamos pasarle el parámetro “`–u`”.

A continuación mostramos un ejemplo de este fichero de configuración, preparado para arrancar un sistema GNU/Linux y otro Windows™:

```
lba32
boot = /dev/hda
prompt
timeout = 50
message = /etc/message
default = debian
restricted
password = contraseña

    image = /vmlinuz
    label = debian
    root = /dev/hda1

    other = /dev/hda3
    label = w2000
```

ANOTACIONES

Toda esta configuración sirve para un arranque estándar del sistema. También hay otro tipo de arranque que utiliza una imagen llamada de RAM Disk (**initrd**). Este otro tipo de arranque sirve para realizar una configuración modular del núcleo Linux. Es muy utilizado cuando necesitamos un núcleo con alguna configuración especial, para incluir módulos en el mismo núcleo, para realizar una imagen de arranque para un CD*live*, para tener una misma imagen para todos los ordenadores de un laboratorio almacenada en un único servidor, etc. De todas formas, las instalaciones estándar del sistema operativo no utilizan casi nunca este tipo de arranque. Si queremos crearnos una imagen de este tipo, podemos informarnos en el manual de `initrd` y en el del programa `mkinitrd`.

6.3.2. Grub

Contenido complementario

El Grub (*GRand Unified Bootloader*) es el programa de arranque del proyecto GNU.

Igual que el Lilo, el Grub también nos sirve para instalar un programa en la zona de arranque que queramos del ordenador. A diferencia del Lilo, tiene muchísimas más posibilidades que lo hacen muy versátil: permite tener un pequeño intérprete de comandos al arrancar el ordenador, nos permite acceder a los archivos de las particiones del disco sin cargar ningún operativo, etc. Como en el caso anterior, en esta sección sólo veremos su configuración básica. Si quisiéramos profundizar más en su uso, podemos recurrir a su manual o en el HOWTO correspondiente.

El sistema de arranque del Grub se carga en dos fases. Generalmente, con la instalación del paquete, se incorporan dos ficheros correspondientes a estas dos fases. Si queremos que al arrancar el Grub no nos muestre ningún menú para seleccionar el operativo que queremos cargar, sólo debemos ejecutar el programa `Grub` y ejecutarlo en el intérprete de comandos que nos muestra:

```
$ install (hd0,0)/PATH/stage1 d (hd0) (hd0,0)/
PATH/stage2
```

Esta instrucción instala el Grub en la MBR del disco maestro del primer canal IDE. La forma como se referencian los discos varía un poco de como se hace en GNU/Linux y con el Lilo. En "`hdX`" la "`X`", en lugar de "`a`", "`b`",. . . , es "`0`", "`1`", etc. Para las particiones también se empieza con el número "`0`" para denominar la primera y a

ANOTACIONES

diferencia de `hda1`, se debe escribir `(hd0,0)` y consecutivamente para las otras. Leyendo la instrucción de esta forma fijémonos cómo el primer parámetro sirve para designar dónde está el archivo de la primera fase del Grub (le indicamos la partición correspondiente, directorio –`PATH` y fichero -`stage1`-). Generalmente, cuando instalamos el paquete del Grub también se añaden estos dos ficheros para cada una de las fases de carga (suelen estar situados en `/usr/share/grub/i386-pc/`). El parámetro `d (hd0)` indica que la primera fase del Grub se instalará en la MBR del primer disco. La última opción especifica dónde está situado el fichero para la segunda fase de carga, que es ejecutada por la primera.

Con esta configuración, al reiniciar el ordenador aparecerá, por defecto, el intérprete de comandos del Grub. Con él podemos manipular muchos aspectos del disco, arrancar el sistema operativo que queremos, etc. Si deseamos arrancar un sistema GNU/Linux escribiremos las siguientes instrucciones:

```
$ kernel (hd0,0)/vmlinuz root=/dev/hda1 $ boot
```

Con la primera indicamos dónde está situada la imagen núcleo (con los parámetros que queramos) y con la segunda iniciamos el proceso de carga del operativo. Si optamos por un menú de selección para no tener que escribir estos comandos cada vez que arrancamos el ordenador podemos generar un fichero de menú como el siguiente (los comentarios empiezan por "#"):

```
#Especificación del operativo que se cargará por
defecto. #Este número está en correspondencia con
el orden de los #sistemas que hay en las secciones
locales a los operativos. default 0

#Indicamos que espere 10 segundos antes de cargar
el sistema #configurado por defecto.
timeout 10

#Configuración de arranque para un sistema GNU/Li-
nux title Debian GNU/Linux
kernel (hd0,0)/vmlinuz root=/dev/hda1
```

Nota

Para ver todos los comandos disponibles en el *shell* del Grub podemos apretar TAB. También se incluyen ayudas para tener una referencia completa de todos los comandos.

Nota

Otra forma de instalar el grub es utilizando el programa grub-install.

ANOTACIONES

```
#Configuración de arranque para un sistema Windows
title W2000
root (hd0,2)
makeactive
```

Para instalar el Grub con este menú de arranque, deberíamos ejecutar la misma instrucción que anteriormente pero añadiendo el parámetro `p (hd0,0)/PATH/menu.lst` con el disco, camino y fichero de menú. Para proteger el sistema de arranque (que desprotegido es aún más peligroso que con el Lilo) podemos poner la directiva de *`password`* `CONTRASEÑA` en la sección global del fichero de configuración. De esta forma, cuando desde el menú se quiera entrar en el *shell* del Grub se pedirá la contraseña. Como en el caso del Lilo, si utilizamos esta directiva es muy importante que sólo el *root* pueda leer este archivo de configuración (aunque en este caso también existe la opción de poner la contraseña cifrada con MD5).

6.4. Acceso a otras particiones y dispositivos

Los sistemas tipo UNIX tratan todos los dispositivos del ordenador como si fueran ficheros. Esto nos permite mucha flexibilidad, ya que podemos aprovechar todos los mecanismos y funciones que utilizábamos con los ficheros y aplicarla a los dispositivos. En el directorio `/dev/` tenemos todos los dispositivos reconocidos por el sistema. Si el sistema no reconoce adecuadamente un dispositivo o queremos crear uno especial, el comando `mknod` nos permite realizar esta clase de operaciones, aunque es importante saber exactamente qué queremos hacer antes de utilizarlo, ya que su mal uso podría dañar partes del sistema.

Para las unidades de almacenamiento, el sistema nos proporciona otro tipo de operación para poder acceder a sus sistemas de archivos, la de **montaje**. Para esta operación utilizaremos los comandos `mount` y `umount`, que sitúan (montan) o desmontan todo el sistema de ficheros de un determinado dispositivo/unidad en un directorio existente del sistema. La forma básica de utilizar el comando es "`mount dispositivo directorio`", donde `dispositivo` puede referenciar cualquier dispositivo del canal IDE o SCSI (`/dev/hdXX`, `/dev/sdXX`), la disquetera (`dev/fdX`), cintas de *backup*, etc.

Contenido complementario

Para la disquetera y CD/DVD muchas distribuciones ya crean un directorio por defecto donde montarlos (`/floppy/` o `/mnt/floppy/` y `/CD-ROM/` o `/mnt/CD-ROM/`). También se suele proporcionar el directorio `/mnt/`, donde podemos crear directorios para otros dispositivos que tengamos en el sistema.

y `directorio` es la ubicación donde montaremos la estructura de ficheros del dispositivo. Es recomendable que el directorio donde montemos estos dispositivos esté vacío, ya que cuando se utiliza como punto de montaje no se puede acceder a ellos.

Para desmontar uno de estos dispositivos podemos utilizar "`umount directorio`", donde directorio debe ser el punto de montaje utilizado. Si montamos dispositivos como un disquete o CD, es importante no sacar el dispositivo del soporte, ya que antes debemos avisar al sistema para que actualice la caché del sistema de ficheros del dispositivo. Igualmente, tampoco podemos desmontar el dispositivo si algún usuario o aplicación está utilizando alguno de sus archivos o directorios (al intentarlo, el sistema daría un mensaje de error).

Fijémonos que con el comando de montaje no estamos especificando en ningún momento el tipo de sistema de ficheros utilizado en la unidad que queremos montar, de forma que se deberá determinar de forma automática. Si queremos especificarlo manualmente, podemos pasar al comando `mount` el parámetro "`–t tipo`" donde tipo podría ser alguno de los de la siguiente tabla (ir al manual de `mount` para ver el listado completo):

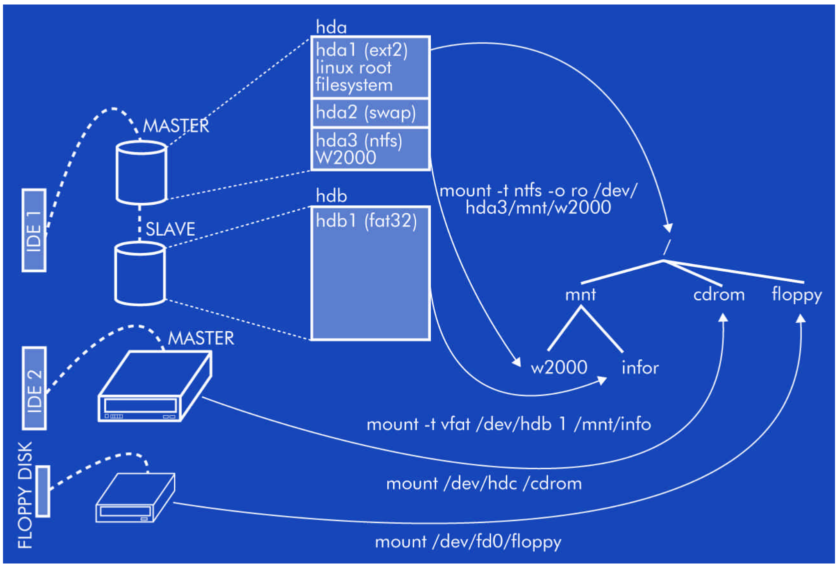
TIPO	SISTEMA
ext	GNU/Linux (versiones de núcleo anteriores a 2.1)
ext2	GNU/Linux (versiones de núcleo posteriores a 2.1)
ext3	GNU/Linux (versiones de núcleo posteriores a 2.2 o 2.4)
swap	Sistema de swap de GNU/Linux
sysv	Sistemas tipo UNIX
minix	MINIX
iso9660	Sistema de ficheros que utilizan la mayoría de CD
nfs	Sistema de ficheros remoto (*Network File System*)
smbfs	Sistema de ficheros remoto en redes Windows™(*Samba File System*)
ntfs	Rama de WindowsNT™
msdos	MS-DOS™
vfat	Rama de Windows95™

ANOTACIONES

Además de pasar el tipo de sistema de ficheros utilizado por la unidad que queremos montar, también podemos indicar otras opciones que nos pueden ser muy útiles en determinadas situaciones (siempre precedidas por "-o" y, si queremos pasar más de una, separadas por comas):

significado de la opción	permitimos	no permitimos
Ejecución de binarios	`exec`	`noexec`
Uso del bit de SetUserId	`suid`	`nosuid`
Ficheros de sólo lectura	`ro`	`rw`
Sistema sincronizado (uso de caché de disco)	`sync`	`async`
Interpretación de caracteres o bloques especiales	`dev`	`nodev`
Permiso para que cualquier usuario monte o desmonte el dispositivo	`user`	`nouser`
Sistema de tipo swap	`sw`	
(si pasásemos `defaults` se utilizarían las opciones `rw`, `suid`, `dev`, `exec`, `auto`, `nouser` y `async`)		

En la siguiente figura podemos ver un ejemplo de utilización de este comando para montar varios dispositivos diferentes:



Además de estos comandos para montar y desmontar unidades, el sistema operativo nos proporciona otro modo de hacer lo mismo y tener siempre una determinada configuración según la unidad. En el fichero `/etc/fstab` podemos guardar esta información de forma

que cada línea indicará una unidad con su directorio de montaje y las opciones que queramos configurar. La sintaxis de cada una de estas líneas será:

```
<disp> <dir> <tipoSistema> <opciones> <dump> <orden>
```

En el primer campo debemos especificar el dispositivo tal como hacíamos con el comando de `mount` y el segundo será el directorio donde queremos montar la unidad indicada. En el campo de `tipoSistema` podemos especificar el sistema de ficheros que utiliza la unidad o bien `auto` para que lo detecte automáticamente. En `opciones` podemos escribir las mismas que utilizábamos con el comando de `mount`, separadas por comas si ponemos más de una. Una opción útil de este campo es la configuración de `auto` o `noauto`, con lo cual indicamos al sistema que monte automáticamente (o no) la unidad al arrancar. El campo de `dump` indica si queremos realizar copias de seguridad (más información en el manual de `dump`). Si no utilizamos este sistema, podemos poner un "`0`". El último campo sirve para indicar el orden de montaje de las unidades. Si le ponemos un "`0`", indicamos que el orden no es importante. La raíz del sistema de ficheros es lo primero que se debe montar, con lo cual en este campo debería haber un "`1`".

Nota

Con el comando "`mount -a`" se montarían todos los dispositivos que tuviesen la opción de `auto` activada (que está por defecto) de este fichero de configuración.

Una entrada que siempre veremos en este fichero y que nos puede sorprender es la del directorio `/proc/`, que tiene un significado especial. Realmente, lo que hay en este directorio no son ficheros, sino el valor de muchas de las variables que utiliza el núcleo del sistema. Siguiendo la misma política del operativo, con la cual todo se debe poder referenciar como un archivo, en el directorio `/proc/` también podemos manipular variables internas del núcleo como si se tratara de ficheros.

Aunque todo este diseño para montar las unidades en la estructura jerárquica de directorios es muy potente y nos permite mucha flexibilidad, en algunos casos no es muy práctico. Por ejemplo, cada vez que queremos copiar un archivo a un disquete tendremos que montar la unidad, copiar y desmontarla de nuevo. Por esta razón, existen algunas otras aplicaciones que nos facilitan todo este proceso para ahorrarnos algunos pasos. Una de ellas son las `mtools`, que es un paquete con varias aplicaciones que nos permiten copiar directa-

ANOTACIONES

mente archivos a un disquete y desde un disquete, y algunas otras herramientas interesantes. También existe un paquete llamado `autofs`, que detecta automáticamente la inserción de algún dispositivo en el sistema y los monta sin necesidad de escribir ningún comando.

6.5. Configuración de dispositivos

Aunque en los inicios de GNU/Linux no era así, actualmente cada vez más fabricantes proporcionan *drivers* para sus dispositivos especiales de GNU/Linux. Antes de intentar configurarlos, debemos buscar información sobre ellos en los mismos manuales del sistema, los módulos, en Internet, etc. para ahorrarnos problemas en su puesta en marcha. Aunque actualmente la mayoría de dispositivos cuentan con HOWTOS, manuales o algún tipo de documentación, es importante que antes de comprar uno nuevo, nos informemos adecuadamente de si existe algún *driver* disponible para él.

Aunque en esta sección sólo veremos cómo configurar algunos de los dispositivos más frecuentemente utilizados en los ordenadores personales, según la distribución que utilicemos, el proceso puede variar significativamente. En toda la sección nos hemos basado en el sistema que se utiliza en Debian GNU/Linux, aunque si aprendemos desde la base cómo funciona este proceso, no deberíamos tener problemas para buscar cuál es la configuración que se utiliza en otras distribuciones.

6.5.1. El teclado

La correcta configuración del teclado es un aspecto muy importante para poder solucionar cualquier problema que nos surja con él. En primer lugar, debemos saber que cuando el sistema arranca, se carga un mapa de caracteres correspondiente al configurado en el proceso de instalación. Este mapa de caracteres se suele encontrar situado en `/etc/console/boottime.kmap.gz` o en algún otro directorio de `/etc/`. Si cambiásemos de teclado, sólo tendríamos que cambiar este fichero con el que correspondiera al nuevo teclado y al arrancar de nuevo ya se cargaría el nuevo mapa. Todos los mapas de caracteres existentes se suelen situar dentro del directorio /

ANOTACIONES

usr/share/keymaps/ ordenados por arquitecturas de ordenadores y países. Concretamente, el que utilizamos en España con ordenadores basados en la arquitectura i386 y un teclado estándar lo encontraríamos en i386/qwerty/es.kmap.gz.

Aunque todos estos mapas de caracteres están comprimidos en formato gzip, podemos descomprimirlos y cambiar alguna de sus entradas para adaptarlas a nuestras necesidades. En cada línea encontramos la directiva keycode, que indica que estamos definiendo una tecla, indicada en el número que sigue a la directiva (podemos saber qué número corresponde con cada una de las teclas a partir del comando showkey). Después de esta definición, tenemos los significados que tiene la tecla apretándola sola, con el SHIFT, etc.

Veámoslo con un ejemplo:

```
keycode 53 = minus underscore control keycode 53 =
Delete
```

En la primera línea se indica qué carácter corresponde al apretar la tecla sola (*minus*) o con el SHIFT apretado (*underscore*). La segunda nos muestra la función que se realizará en caso de apretar la tecla juntamente con la de CTRL (se eliminaría el siguiente carácter). Toda la información necesaria para configurar correctamente un archivo de este tipo la podemos encontrar en el manual de *keymaps*, muy útil cuando nos encontramos con algún problema con teclados especiales o de otros países. Si no queremos reiniciar el sistema al cambiar este fichero de mapa, podemos utilizar el comando loadkeys (dumpkeys nos muestra las configuradas).

Otro aspecto relacionado con el teclado es el tema de las diéresis, acentos, etc. Todo ello lo podemos configurar a partir del fichero de /etc/inputrc (todas las directivas posibles de este fichero las tenemos especificadas en el manual de readline). La que nos puede ser más útil es la de convert-meta, que desactivándola (set convert-meta off) nos permite utilizar los acentos y diéresis (para el catalán).

Finalmente, otra configuración importante (indirectamente relacionada con el teclado) es la de **locales**. Con locales podemos configu-

Contenido complementario

Con el programa consolechars podemos cargar la fuente que queremos en el terminal. Debemos diferenciar claramente lo que es una fuente y lo que es el mapa de caracteres: el mapa nos determina qué significado tiene cada tecla, mientras que la fuente es sólo la representación gráfica que le damos a la misma. Toda la configuración de las fuentes de caracteres se suele encontrar en /etc/console-tools/config.

ANOTACIONES

rar la zona o zonas geográficas en las que estamos para poder utilizar teclas especiales del teclado, ver las fechas en el formato al que estamos acostumbrados, etc. Esta configuración es utilizada por muchas de las librerías del sistema, de forma que en muchos comandos y aplicaciones del sistema se utilizará su configuración para adaptar algunas funciones a nuestro entorno local. Su configuración la podemos encontrar en `/etc/locale.gen` y podemos utilizar los comandos `locale-gen` y `locale` para verla o actualizarla.

6.5.2. Tarjeta de red (tipo Ethernet)

Para configurar una nueva tarjeta de red (tipo Ethernet), lo primero que debemos hacer es añadir el módulo del núcleo necesario para que se reconozca adecuadamente. Aunque en algunas tarjetas es posible que no tengamos que realizar este paso porque el mismo núcleo ya puede estar compilado para reconocer las más habituales, debemos asegurarnos (antes de comprar la tarjeta) de que existe el *driver* o módulo necesario para ella.

Contenido complementario

Con "`discover -module Ethernet`" podemos saber qué módulo necesita nuestra tarjeta de red. Si queremos dejarlo configurado para que se cargue siempre deberíamos escribirlo en `/etc/modules` (si no, con `modprobe` o `insmode` podemos insertarlo).

Nota

Otra forma de denominar las tarjetas de red es con las siglas de *Network Interface Card* (NIC).

Nota

Otra forma de reconfigurar el keymap y las locales en Debian es utilizado "apt-reconfigure console-data" o "apt-reconfigure locales" respectivamente.

Una vez el sistema reconoce la tarjeta, ya podemos configurarla de la forma que queramos.

En el fichero `/etc/network/interfaces` podemos especificar toda su configuración, donde también tendremos la de las otras interfaces del sistema. Una **interfaz** es un dispositivo (real o lógico) relacionado con la red a partir del cual el sistema se puede comunicar con otros ordenadores, ofrecer unos determinados servicios, etc. Son las puertas que tiene el sistema para poderse comunicar. Para cada interfaz reconocida en el sistema, en este fichero se le especifican las directivas necesarias para su correcto funcionamiento.

Vamos a verlo con un ejemplo:

```
#Interfície de loopback
auto lo
iface lo inet loopback

#NIC
auto eth0
```

ANOTACIONES

```
iface  eth0 inet static
       address 192.168.0.10
       netmask 255.255.255.0
       network 192.168.0.0        #opcional
       broadcast 192.168.0.255    #opcional
       gateway 192.168.0.1        #opcional
```

La primera entrada que encontraremos en este fichero suele ser para la interfaz de *loopback*. Esta interfaz no se corresponde con ninguna tarjeta ni dispositivo real del ordenador, sino que es un mecanismo del operativo que le permite utilizar los protocolos de comunicación de forma interna. De esta forma, si probamos funciones de la red sin comunicarnos con ningún otro ordenador no hace falta ni siquiera tener una tarjeta de red instalada. En todas las entradas encontramos la directiva de `auto` antes de especificar la configuración del dispositivo. Esta directiva indica que la tarjeta se puede montar automáticamente cuando el sistema arranca. La directiva de `iface` especifica el tipo de tarjeta y protocolo que se utilizará con ella por medio de la siguiente sintaxis: "`iface dispositivo familia-Protocolo métodoConfiguración`". Con las tarjetas Ethernet el dispositivo será `ethX`, donde la "`X`" será un número empezando por "`0`", que indica el número de tarjeta instalada en el ordenador. La familia del protocolo de comunicación utilizado con la tarjeta suele ser cualquiera de los siguientes:

- `inet`: IPv4, utilizado en Internet y la mayoría de redes locales.
- `inet6`: IPv6, la nueva versión de IPv4, que poco a poco se va instaurando.
- `ipx`: para redes Novell™.

Nota

A partir de las versiones 2.2 del núcleo Linux, ya se puede utilizar una nueva infraestructura de red denominada **`iproute2`**. Con ella podemos manipular todos los aspectos relacionados con nuestras NIC y las tablas internas que utiliza el operativo para manejar todo lo relacionado con la red.

Finalmente, en el último campo se indica cómo se obtiene la configuración de red de la tarjeta (su dirección, la red dónde está, el *gateway* que hay que utilizar, etc.). En la siguiente tabla podemos ver cuáles son estas opciones para la familia de protocolos `inet`:

CONFIG	OPCIONES	DESCRIPCIÓN
`loopback`		Método para definir la interfaz de *loopback* (se debe utilizar con la interfaz `lo`).
`static`		Método para configurar una NIC con una dirección IP estática.

CONFIG	OPCIONES	DESCRIPCIÓN
	address	Dirección IP de la interfaz. Campo requerido.
	netmask	Máscara de la dirección IP. Campo requerido.
	broadcast	Dirección de *broadcast*. Si no se especifica, se calcula automáticamente.
	network	Dirección de identificación de red. Sólo requerida para versiones del núcleo 2.0.X.
	gateway	Dirección IP del *gateway* que utilizamos para esta interfaz.
dhcp		Método para configurar de forma remota la IP de todos los ordenadores de una red local (*Dynamic Host Configuration Protocol*).
	hostname	IP del servidor de DHCP.
	leasehours	Tiempo, en horas, de alquiler de la IP (pasado este tiempo, se renueva).
	leasetime	Tiempo, en segundos, de alquiler de la IP.
	vendor	Identificador de tipo de servidor (en general, dhcpd).
	client	Identificador del tipo de cliente (en general, dhcpd).
bootp		Método para configurar de forma remota la IP de todos los ordenadores de una red local (BOOT *Protocol*). Actualmente se utiliza más DHCP.
	bootfile	Fichero a utilizar en el momento de arranque.
	server	Dirección IP del servidor de BOOTP.
	hwaddr	Dirección MAC del servidor BOOTP.
ppp		Método utilizado con el protocolo Point to Point Protocol, usado en los módems.
	provider	Proveedor del servicio.

Aunque en este capítulo no entraremos en redes de computadores, debemos saber que disponemos de muchos comandos para manejar la configuración de red del sistema operativo. Los más importantes son `ifconfig`, con el cual podemos ver los dispositivos configurados, `ifdown` e `ifup`, que nos permiten apagar o encender la interfaz que queremos y `route`, que nos muestra la tabla de *routing* del sistema.

6.5.3. Tarjeta WiFi

Las redes de comunicación sin cables son cada vez más frecuentes, tanto en instalaciones domésticas como en instituciones, escuelas o empresas. Para la instalación de las mismas, se debe disponer de lo que denominamos puntos de acceso, que son unos dispositivos conectados a la red física de la institución. Estos puntos de acceso permiten que, a partir de unas tarjetas PCMCIA, cualquier ordenador de

ANOTACIONES

su alrededor pueda conectarse a la red. De esta forma, se simplifica mucho el cableado de los edificios.

Para que nuestro GNU/Linux detecte y configure adecuadamente una tarjeta *wireless* (de tipo PCMCIA) debemos añadir al núcleo del sistema los módulos `orinoco cs` y `hermes`, que en muchos casos ya vienen compilados en el mismo núcleo. Toda la configuración de estas tarjetas se guarda en los ficheros del directorio `/etc/pcmcia/`. Lo único que debemos añadir en el fichero `/etc/pcmcia/config.opts` es (dependiendo de la tarjeta deberíamos cambiar la directiva de "`card`"):

```
card " Conceptronic Wireless"
    version "802.11", "11Mbps Wireless LAN Card"
    bind "orinoco_cs"
```

Con esto ya conseguimos que el sistema reconozca la tarjeta *wireless* como un dispositivo más del sistema. Si no tenemos ninguna otra tarjeta Ethernet instalada, se referenciará como `eth0`, `eth1` si ya tenemos una, etc. A continuación, lo único que nos faltará para que la tarjeta se conecte al punto de acceso será editar el fichero `/etc/network/interfaces` y añadirle la configuración necesaria para que se le asigne una IP. Naturalmente, esta interfaz del sistema la podemos tratar como cualquier otra, utilizando los mecanismos de *firewall* del sistema, con las aplicaciones iproute2, etc.

6.5.4. Módems

Para la configuración de un módem generalmente se suele utilizar la aplicación `pppconfig`, que escribe los archivos de configuración necesarios para el daemon del sistema `ppp`, que es el programa encargado de establecer la conexión a Internet. Con `pppconfig` (o aplicaciones similares) siempre se deben realizar unos determinados pasos, que detallamos a continuación:

1) Nombre del proveedor: el proveedor es la empresa con la que tenemos el contrato de conexión a Internet. Este nombre sirve para poder identificar cada conexión que configuremos de forma única.

Contenido complementario

Los paquetes que contienen los programas necesarios para la conexión a Internet con un módem suelen nombrarse `ppp` y `pppconfig`.

Contenido complementario

Con `pppconfig` tenemos un menú que nos permite añadir, modificar o eliminar conexiones. Los pasos que mostramos se corresponden con la inserción de una nueva conexión.

2) Configuración de servidores de nombres: cuando establecemos el contrato con nuestro proveedor, generalmente se suelen proporcionar la/s IP de los servidores de nombres que deben utilizarse. Si tenemos estas IP, debemos indicar que utilizamos una configuración estática, con lo cual seguidamente se nos pedirán estas IP. Sólo en el caso de que nuestro proveedor nos indique que la configuración de DNS es dinámica, debemos escoger este otro tipo de configuración. Con la tercera opción, que nos informa de que DNS será tratado por otros medios, podemos utilizar la configuración del fichero `/etc/resolv.conf`.

3) Método de autentificación: el método de autentificación puede ser PAP o CHAP. Generalmente, los proveedores suelen utilizar el PAP (*Peer Authentication Protocol*), aunque si no funcionara deberíamos informarnos adecuadamente.

4) Nombre de usuario y contraseña: ésta es la información que nos proporciona el proveedor para poder conectarnos y acceder a sus servicios.

5) Velocidad del módem: según qué módem tengamos, podremos acceder a Internet a mayor o menor velocidad. Actualmente, todos van a 115200 bps, con lo que lo más recomendable es dejar el valor "`115200`". Si tuviéramos un módem más lento, ya se suele detectar y reconfigurar automáticamente en el momento de la conexión.

6) Llamada con pulsos o tonos: la mayoría de centralitas telefónicas ya funcionan con tonos, aunque en determinadas zonas rurales aún se utiliza el antiguo sistema de pulsos.

7) Número de teléfono: este número también debe proporcionarlo el proveedor de Internet.

8) Puerto de comunicación: el puerto de comunicación es el puerto en el cual tenemos conectado el módem. Si le indicamos que lo detecte automáticamente, se realizará un chequeo de todos los puertos y se configurará automáticamente. Si no, podemos indicarlo con `/dev/ttySX`, donde la "`X`" es un 0 para el COM1, un 1 para el COM2, etc.

Toda esta configuración se suele almacenar en los archivos situados en el directorio `/etc/ ppp/`. Aunque también podemos editar estos

ANOTACIONES

ficheros y cambiar las directivas manualmente, es más recomendable utilizar alguna aplicación automática, ya que su configuración es bastante compleja. Para establecer la conexión con nuestro proveedor, deberíamos iniciar el daemon ejecutando "`/etc/init.d/ppp start`". Para pararlo, podemos utilizar "`/etc/init.d/ppp stop`".

6.5.5. Tarjeta de sonido

La tarjeta de sonido necesita la inserción de un módulo del núcleo del sistema para poder funcionar correctamente. Si tenemos instalada la aplicación `discover`, podemos descubrir qué módulo es el que corresponde con nuestra tarjeta por medio del comando "`discover --module sound`". Para instalar el módulo, podemos utilizar los comandos `insmod` o `modprobe`, y si queremos dejarlo configurado permanentemente, deberíamos escribirlo en el fichero `/etc/modules`.

Aunque con la inclusión del módulo correspondiente ya podremos utilizar la tarjeta de sonido adecuadamente, generalmente también se suele instalar la infraestructura de sonido ALSA (*Advanced Linux Sound Architecture*). Generalmente, la mayoría de distribuciones lo suelen incluir por defecto, aunque si no es así, se puede instalar con el paquete correspondiente.

Contenido complementario

ALSA es un proyecto que ha desarrollado mucho software relacionado con aplicaciones de tratamiento de sonido, nuevos módulos para el núcleo Linux, etc.

6.5.6. Impresora

En GNU/Linux, la configuración de impresoras se puede realizar con muchas aplicaciones diferentes. Aunque el `lpd` (Line Printer Daemon) fue uno de los primeros programas de gestión de impresión que aparecieron en los sistemas tipo UNIX, actualmente existen muchísimos más fáciles de configurar y gestionar. A continuación, comentamos algunos de los más utilizados:

- `lpd`: uno de los primeros daemons de impresión de los sistemas tipo UNIX. Su configuración debe realizarse manualmente.
- `lpr`: la versión de BSD del `lpd`. Es muy recomendable utilizar algún tipo de filtro automático como `magicfilter` o `apsfilter`

ANOTACIONES

para configurar las impresoras. Este tipo de filtro detecta automáticamente el tipo de fichero a imprimir y prepara la impresión adecuadamente (utiliza un filtro llamado IFHP).

- `lprng`: aplicaciones basadas en `lpr` con la ventaja que incorpora una herramienta de configuración denominada `lprng-tool`, que permite realizar la configuración de forma gráfica y sencilla.

- `gnulpr`: la versión de GNU del sistema de impresión lpr. También incorpora herramientas gráficas de configuración, gestión de los servicios, etc.

- `CUPS`: de Common UNIX Printing Systems, este conjunto de aplicaciones es compatible con los comandos de `lpr` y también sirve para redes Windows™. Utiliza un conjunto de filtros propios y soporta la gran mayoría de impresoras del mercado.

Aunque todas estas aplicaciones tienen sus propios métodos de configuración, todas utilizan el fichero `/etc/printcap` para guardarla. Generalmente, también utilizan algún tipo de daemon para que el sistema de impresión sea operativo. El daemon se puede configurar para que el ordenador al que está conectada la impresora sirva como servidor de impresión. De este modo, varios ordenadores de la misma red podrán utilizar la misma impresora, ahorrando recursos. Para los clientes de impresión se pueden utilizar los mismos programas especificando, en la configuración, que la impresora es remota (generalmente se debe proporcionar la IP del servidor de impresión y la cola).

Contenido complementario

Al configurar un servidor de impresión, es importante que configuremos adecuadamente desde qué máquinas/usuarios permitimos la impresión. De otro modo, un atacante podría aprovechar la vulnerabilidad y aprovechar nuestros recursos, dejar la impresora sin papel, etc.

Si queremos configurar un servidor de impresión para redes Windows™ o configurar una impresora de un servidor Windows™ desde un cliente GNU/Linux, debemos utilizar otro tipo de programas. `Samba` es un conjunto de aplicaciones de GNU/Linux que utilizan los protocolos de las redes Windows™. Aunque sus funcionalidades van mucho más allá de la configuración de un servidor o cliente de impresión, para poder utilizar impresoras en Windows™ tendremos que utilizar este conjunto de aplicaciones o bien las que nos proporciona `CUPS`.

Contenido complementario

`Swat` (*Samba Web Administration Tool*) es una herramienta muy útil para la configuración de un servidor de samba.

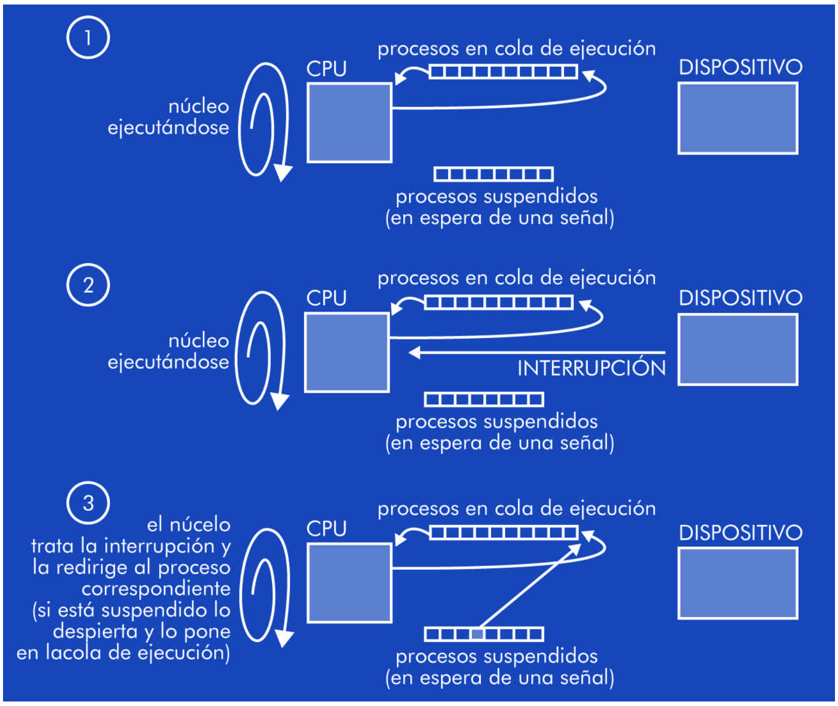
7. Daemons y *runlevels*

7.1. Los daemons

Como ya sabemos, GNU/Linux nos permite ejecutar simultáneamente tantos procesos como queramos repartiendo equitativamente el tiempo de la CPU entre ellos. De hecho, el mecanismo de manejo de procesos también debe tener en cuenta lo que se llaman **interrupciones**. Una interrupción es una señal que llega al núcleo del sistema desde cualquiera de los dispositivos que tenemos instalados en nuestro ordenador. Estas interrupciones suelen estar vinculadas a algún proceso en concreto, de forma que el núcleo debe despertar el proceso en cuestión (si no está en ejecución) y redirigirle la interrupción para que la procese adecuadamente. Un ejemplo típico de interrupción es cuando apretamos una tecla del teclado o movemos el ratón: al hacerlo, el dispositivo envía una señal que debe ser redirigida hacia la aplicación correspondiente para que sea tratada de forma adecuada.

Para poder manejar adecuadamente todas las interrupciones que se producen, el núcleo no escucha permanentemente a los dispositivos del sistema esperando sus señales. En lugar de hacerlo, el sistema ejecuta las operaciones de los procesos en cola de ejecución y sólo cuando se produce una interrupción atiende al dispositivo que la ha generado. Esto debe realizarse de esta forma debido a la gran diferencia de velocidad entre los dispositivos del sistema y la CPU. El tratamiento de interrupciones es fundamental para cualquier sistema operativo, ya que es este mecanismo, entre otros, el que nos permite mantener en ejecución tantos procesos como queramos y, en cuanto lleguen las interrupciones, despertar los procesos que las están esperando.

ANOTACIONES



Un **daemon** (*Disk And Execution MONitor*) es un proceso que, generalmente, tenemos cargado en memoria, esperando alguna señal (proveniente de una interrupción de dispositivo o del mismo núcleo) para despertarse y ejecutar las funciones necesarias para tratarla. Aunque esta definición también puede encajar con otros procesos del sistema (lanzados por los usuarios o por el mismo sistema), un daemon también suele ajustarse a esta forma de ejecución (aunque en algunos casos especiales, no). De esta forma, los daemons que tengamos cargados no ocupan la CPU mientras no es estrictamente necesario y por muchos que tengamos en memoria siempre podremos trabajar con el ordenador sin problemas.

Aunque un daemon sea un proceso como cualquier otro que se ejecuta en modo *background*, la forma como los organizamos y tratamos sí que es diferente del resto de comandos y programas del sistema. Generalmente, todos los daemons tienen un *shell script* situado en el directorio `/etc/init.d/` que nos permite iniciarlo, pararlo o ver su estado de ejecución. Para realizar algunas de estas funciones debemos ejecutar el *shell script* correspondiente al daemon que queramos tratar pasándole alguno de los siguientes parámetros:

- `start`: para iniciar el daemon. Si éste ya estuviera ejecutándose, se muestra un mensaje de error.

Contenido complementario

Los *shell scripts* de los daemons no son más que una herramienta para facilitar todo su proceso de arranque, parada, etc. En algunos casos, también podemos utilizar el mecanismo y organización de estos daemons para poder ejecutar ciertas operaciones que nos interesen (escribiendo un *shell script* que se ejecute al entrar en un determinado nivel de ejecución).

ANOTACIONES

- `stop`: para parar el daemon. Si no estuviera ejecutándose, se muestra un mensaje de error.

- `restart`: reinicia el daemon. Sirve para que se vuelvan a leer los archivos de configuración del mismo.

- `reload`: aunque no todos los daemons lo permiten, este parámetro sirve para poder recargar los archivos de configuración sin tener que pararlo.

La mayoría de estos *scripts* utilizan un programa llamdo `start-stop-daemon` que nos proporciona el sistema operativo y que sirve para el tratamiento de estos procesos. Es habitual que al administrar un servidor tengamos que diseñarnos nuestros propios daemons para realizar alguna tarea concreta. En el directorio donde se sitúan todos los *shell scripts* de los daemons también se suele encontrar uno de ejemplo (`/etc/init.d/skeleton`) para que lo podamos utilizar cuando necesitemos configurar uno nuevo que no esté en la distribución. Generalmente suelen estar programados de la siguiente manera:

```
#! /bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:
    /usr/sbin:/usr/bin
    DAEMON=/usr/sbin/daemon
    NAME=daemon

DESC="some daemon"

test -x $DAEMON || exit 0
set -e
case "$1" in
  start)
    echo -n "Starting $DESC: $NAME"
    start-stop-daemon --start --quiet --pidfile \
      /var/run/$NAME.pid --exec $DAEMON
      echo "."
;;
stop)
   echo -n "Stopping $DESC: $NAME "
```

Contenido complementario

Para ejecutar un daemon, debemos llamarlo con su ruta completa (`/etc/init.d/ nombreDaemon`) y pasarle el parámetro que nos interese. Algunas distribuciones incorporan el comando service, que permite hacer lo mismo sin tener que especificar la ruta completa.

ANOTACIONES

```
   start-stop-daemon --stop --quiet --pidfile \
    /var/run/$NAME.pid --exec $DAEMON
   echo "."
;;
restart|force-reload)
   echo -n "Restarting $DESC: $NAME"
   start-stop-daemon --stop --quiet --pidfile \
   /var/run/$NAME.pid --exec $DAEMON
sleep 1
start-stop-daemon --start --quiet --pidfile \
   /var/run/$NAME.pid --exec $DAEMON
   echo "."
;;
*)
   N=/etc/init.d/$NAME
   echo " Usage: $N {start|stop|restart|force-re-
load}" >&2
   exit 1
   ;;
esac
exit 0
```

Contenido complementario

Aunque los daemons son programas como cualquier otro, su programación difiere un poco de las aplicaciones de usuario porque deben incluir funciones para quedar suspendidos y esperar señales para que sean despertados, etc.

Como hemos visto, en las variables declaradas al inicio del *shell script* especificamos qué `PATH` es necesario para el proceso del daemon, el programa a ejecutar (`DAEMON`), el nombre que le damos (`NAME`, que debe ser igual que el nombre del *shell script*) y su descripción (`DESC`). Lo único que hace el código al arrancar el daemon es escribir en el directorio `/var/run/` un fichero con el PID del proceso. Al pararlo, se va a buscar este PID y se envía la señal de finalización al proceso correspondiente. Naturalmente, encontraremos *shell scripts* preparados para realizar muchísimas más operaciones con el daemon a tratar, aunque como mínimo todos deben tener esta estructura.

7.2. Los *runlevels*

Los daemons que tengamos ejecutándose en un determinado momento nos marcan los servicios que el sistema operativo está ofreciendo y/o recibiendo. El hecho de que podamos tener tantos daemons diferentes hace que tengamos que plantear su organización de forma

ANOTACIONES

adecuada. Entenderemos un *runlevel* (o '**nivel de ejecución**') como la ejecución de unos determinados daemons que a su vez proporcionan unos servicios concretos. En la instalación de un servidor es habitual diseñar una configuración para que en determinados momentos se puedan ofrecer determinados servicios y en otros no. Para permitir este tipo de funcionamiento, el sistema operativo nos proporciona diferentes niveles de ejecución que podremos adaptar a nuestras necesidades.

Si bien podemos configurar el número de niveles de ejecución que queremos y la funcionalidad de cada uno de ellos, generalmente los sistemas like UNIX nos proporcionan 6 diferentes con las siguientes propiedades:

NIVEL	Funcionalidad
0	El nivel de ejecución 0 está configurado para parar el sistema.
1	Este nivel es denominado como *single user*, ya que sólo permite la entrada al sistema al *root* del mismo. Se arrancan los daemons mínimos y sirve para tareas de mantenimiento.
2~5	Los niveles del 2 al 5 están destinados para ser configurados según las necesidades de cada instalación. Al instalar el sistema, por defecto todos son iguales. Estos niveles también se llaman multiusuario, ya que, por defecto, permiten que más de un usuario trabaje en el sistema.
6	El último nivel está preparado para reiniciar el sistema. Es muy parecido al 0 pero se añade una función de reinicio.

El comando necesario para cambiar de nivel de ejecución es `init` (le pasamos como parámetro el nivel de ejecución que queramos) y para ver en cuál estamos, `runlevel`.

Los comandos `halt`, `reboot`, `shutdown` o `poweroff` lo único que hacen es llamar al nivel de ejecución 0 o 6 realizando, antes, alguna operación concreta (ver su manual para más información). Sólo el *root* del sistema puede utilizar todos estos comandos.

La forma como se organizan estos daemons en cada nivel de ejecución es muy simple. Cada nivel de ejecución tiene un directorio situado en `/etc/rcX.d/` donde la "`X`" es el número de nivel. En estos directorios encontramos enlaces simbólicos a los *shell scripts* de los daemons situados en `/etc/init.d/`, que nos sirven para indicar al sistema si queremos iniciar o parar el daemon al que apuntan. Con el mismo nombre del enlace se identifica la acción a realizar: si el enlace empieza por "`S`" (*Start*) indicamos que queremos iniciar el

ANOTACIONES

daemon, mientras que se empieza por "`K`" (*Kill*) indica que queremos pararlo. Si el nombre no empieza por ninguna de estas letras, el sistema no hace nada con él. Después de esta letra se pone un número de 2 cifras entre "`00`" y "`99`", que indica el orden de inicio o parada de los mismos. Este orden es importante, ya que algunos daemons necesitan que otros estén en ejecución antes de ser iniciados.

Al cambiar de nivel de ejecución, el sistema inspeccionará los daemons del directorio correspondiente y empezará, primero, parando los daemons indicados y después iniciará los demás. Lo único que se hace es llamar al daemon pasándole como parámetro `start` o `stop`, de forma que si paramos alguno que no se esté ejecutando en el momento de parada, no pasaría nada porque el mismo *shell script* lo tiene en cuenta. Esto nos sirve para poder cambiar de nivel de ejecución sin tener en cuenta el nivel anterior al que estábamos. En la siguiente figura podemos ver un ejemplo de configuración para 3 niveles de ejecución:

NIVEL DE EJECUCIÓN 2		**DAEMONS EJECUTÁNDOSE**
K50sshd K51apache-ssl K52tftpd K53telnet	S10sysklogd S12kerneld S20dhcpd S50proftpd S90apache	*sysklogd* *kerneld* *dhcpd* *proftpd* *apache*
NIVEL DE EJECUCIÓN 3		
K50dhcpd K51proftpd K52apache K53tftpd K53telnet	S10sysklogd S12kerneld S20sshd S50apache-ssl	*sysklog* *kerneld* *sshd* *apache-ssl*
NIVEL DE EJECUCIÓN 4		
K50dhcpd K51proftpd K52apache K53tftpd K53telnet	S10syslogd S12kerneld S20tftpd S50telnet	*sysklogd* *kerneld* *tftpd* *telnet*

En el fichero `/etc/inittab` tenemos definida toda la configuración de los *runlevels*: el nivel de ejecución por defecto, el número de consolas disponibles en cada uno de ellos, etc. Cada línea del fichero es una directiva con la sintaxis: "`<id> : <runlevels> : <action> : <process>`". El primer campo es el identificador de la directiva, seguidamente encontramos en qué niveles de ejecución es válida esta

directiva, la acción a realizar y el proceso a lanzar. En el siguiente ejemplo explicamos cómo configurar algunas de estas directivas:

```
# El nivel de ejecución por defecto (en este caso,
el 2) id:2:initdefault:

# Scripts a ejecutar al arrancar el sistema (antes
# de entrar en el nivel de ejecución por defecto)
si::sysinit:/etc/init.d/rcS

# Programa que se llama al entrar en el nivel de
ejecución
# single user (la acción wait indica que se lanza el
# proceso y no se hace nada más)
~~:S:wait:/sbin/sulogin

# Configuración de los diferentes niveles de eje-
cución
# disponibles en el sistema
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6

# Comando a ejecutar al apretar CTRL+ALT+DEL
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

# Definición de las consolas abiertas en cada
# nivel de ejecución (la acción respawn indica
# que al terminar la ejecución del proceso
# getty se lance otra vez)

1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
4:23:respawn:/sbin/getty 38400 tty4
5:23:respawn:/sbin/getty 38400 tty5
6:23:respawn:/sbin/getty 38400 tty6
```

ANOTACIONES

Como vemos, en este fichero se configura todo lo referente a los niveles de ejecución de forma muy flexible pudiendo cambiar lo que nos interese para adaptarlo mejor a nuestras necesidades. Fijémonos que, aunque aquí definamos el nivel de ejecución por defecto, también lo podríamos especificar al arrancar el sistema con el Lilo o Grub. Esto es muy útil, por ejemplo, cuando tenemos problemas graves en el sistema que no nos permiten arreglarlos adecuadamente; si arrancamos con el primer nivel (pasando "`1`" o "`single`" al Lilo o Grub), sólo se iniciarán las funciones más necesarias y podremos entrar para arreglar lo que haga falta.

Contenido complementario

En este fichero también podríamos configurar un terminal que se comunicara con el sistema a partir de un módem y otro ordenador con la directiva "`T1:23:respawn:/sbin/ mgetty -x0 -s 57600 ttyS1`". De esta forma, podríamos tener una consola del sistema en otro terminal comunicándonos con una línea telefónica.

7.3. El arranque del sistema

El proceso padre de todos los demás es el `init`. Este proceso se encarga de arrancar los otros que tengamos en el nivel de ejecución configurado. Sin embargo, antes de entrar en este nivel se ejecutan todos los *shell scripts* de `/etc/rcS.d/` (configurado en `/etc/inittab`), que pueden ser o bien daemons como los de los otros *runlevels* o simplemente *shell scripts* necesarios para el sistema (carga del mapa de caracteres, carga de los módulos del núcleo, etc.). Si queremos eliminar alguno de ellos, debemos saber exactamente qué estamos haciendo, ya que generalmente son imprescindibles para el buen funcionamiento del operativo.

Una vez se han arrancado estos daemons (o *shell scripts*), se entra en el nivel de ejecución configurado por defecto, parando e iniciando los daemons especificados en él. Una vez aprendida toda esta organización, ya podremos adaptar el arranque del sistema a nuestras necesidades creando y situando los daemons que queramos en cualquiera de los sitios que hemos visto.

7.4. Daemons básicos

Según la distribución de GNU/Linux que utilicemos, el mismo proceso de instalación ya configura unos daemons u otros. Aun así, todas las distribuciones suelen incorporar el daemon para el sistema de logs y el de la ejecución periódica y retardada de aplicaciones (aun-

ANOTACIONES

que las configuraciones de los mismos pueden variar un poco). En esta sección veremos cómo funcionan estos tres daemons básicos y cómo podemos configurarlos. Es importante saber manejar estos daemons básicos porque nos pueden ayudar mucho en algunas de las tareas de administración.

7.4.1. Logs de sistema (sysklogd)

Los logs del sistema son ficheros de traza que un daemon del operativo se encarga de generar para que quede constancia de cualquier acción realizada sobre el mismo. El daemon encargado de realizar estas tareas es el `sysklogd`, cuya configuración encontramos en `/etc/syslog.conf`. Cada línea de este fichero consiste en una regla con dos campos: el selector y la acción. Con el selector configuramos de qué servicio queremos tratar los logs y el nivel de prioridad de los mismos. La acción sirve para indicar hacia dónde queremos redirigir los logs (a un fichero, a una consola, etc.). En las tablas de la siguiente página podemos ver las diferentes opciones válidas para estos campos.

Contenido complementario

El núcleo del sistema también lanza un daemon para gestionar sus logs denominado `klogd`.

Generalmente, todos los ficheros de logs del sistema se suelen almacenar en el directorio `/var/log/`. Aunque la mayoría de ficheros de logs son de texto y los podemos ver con cualquier editor, podemos encontrar alguno especial que no guarde sus datos en este formato. Generalmente suelen ser los ficheros `/var/log/wtmp` y `/var/log/btmp`, que son los logs de entrada de usarios en el sistema y de entradas erróneas respectivamente. Para ver estos dos ficheros, podemos utilizar los comandos `last` y `lastb`. Si tuviéramos configurados estos logs en algún otro fichero, también podríamos verlos pasando el parámetro "`-f fichero`" al comando `last`.

Contenido complementario

Para ver los últimos registros de entrada de los usuarios, también podemos utilizar el comando `lastlog`.

SELECTOR			
Servicio	*significado*	Prioridad	*significado*
authpriv	Mensajes de autorizaciones o de aspectos de seguridad.	emerg	El sistema es inutilizable.
cron	Daemon crond y atd.	alert	La acción se debe realizar de inmediato.
daemon	Daemons del sistema sin opciones de logs.	crit	Condiciones críticas.
ftp	Daemon del servidor FTP (File Transfer Protocol).	err	Condiciones de error.

SELECTOR			
Servicio	*significado*	Prioridad	*significado*
kern	Mensajes del núcleo del sistema.	warning	Condiciones de emergencia.
lpr	Mensajes del subsistema de impresión.	notice	Noticias normales, pero importantes.
mail	Mensajes del subsistema de correo (si lo tenemos configurado).	info	Mensajes de información.
news	Mensajes del subsistema de noticias (si lo tenemos configurado).	debug	Mensajes de *debugging*.
syslog	Logs generados por el mismo daemon syslogd.		
user	Logs de aplicaciones de nivel de usuario.		
uucp	Mensajes generados por el sistema de UUCP (Unix-To-Unix Copy Protocol).		
local0~7	Reservados para su uso local.		
(los servicios y prioridades se pueden combinar como se quiera)			

ACCION	
Destino	*explicación*
Fichero regular	Se especifica la ruta completa del fichero. Poniendo un "–" delante no se requiere que el fichero sea sincronizado cada vez que se escribe en él (aunque se perderá el log en caso de fallar la alimentación).
Pipe nombrado	Este sistema permite que los mensajes se redirijan hacia una tubería creada antes de iniciar el daemon de `sysklogd` con el comando `mkfifo`. Se indica poniendo el carácter "`|`" antes del nombre del fichero. Es muy útil para operaciones de *debugging* de programas.
Consola	Especificando `/dev/ttyX` dónde "X" es un número de consola o `/dev/console` los logs se redirigen a la pantalla especificada.
Máquina remota	Para especificar que los logs se redirijan a una máquina remota, debemos preceder el nombre del host remoto con "`@`".
Usuarios	Especificando el nombre de usuario o usuarios (separados por comas) los logs correspondientes se redirigen a éstos.
Usuarios on-line	Con "`*`" especificaremos que los logs se redirijan a todos los usuarios que en el momento de ocurrir el log estén dentro del sistema. Esto se utiliza para avisar a todos los usuarios que ha pasado alguna acción crítica en el sistema.
(las acciones se pueden poner en todos los selectores que se quiera)	

NOTACIONES

Esta forma de tratar los logs permite mucha flexibilidad para configurarlos adecuadamente cuando instalamos un servidor, tarea muy importante para tener controlados los aspectos que más nos interesan del sistema. Aun así, si tuviéramos que guardar todos los logs que se generan en un servidor, seguramente al final saturaríamos el disco por el tamaño siempre creciente de estos archivos. Para evitarlo se utiliza un sistema de rotación de logs, que consiste en ir compri-

miendo, cada cierto tiempo, estos ficheros y guardar sólo hasta una determinada antigüedad. Aunque generalmente se suelen comprimir cada semana y se guardan sólo los de uno o dos meses anteriores, podemos configurar todo esto a partir del fichero /etc/logrotate.conf. Los logs de ciertos servidores y/o aplicaciones también se pueden configurar de forma explícita para tener un control más adecuado de lo que hacen. La configuración personalizada de logs para estas aplicaciones suele situarse en /etc/logrotate.d/.

Internamente, el sistema utiliza unos programas para manejar de forma más amena todo este sistema de logs. Con logger podemos escribir en el sistema de logs del sistema. savelog y logrotate sirven para guardar y, opcionalmente, comprimir algunos de los ficheros de logs que tenemos (con el segundo podemos configurar más opciones que con el primero). Estos comandos también se pueden utilizar para crear nuestros propios ficheros de logs o, si es necesario, manipular manualmente los del sistema (con el manual de los mismos obtendremos más información sobre su tratamiento y manipulación).

7.4.2. Ejecuciones periódicas (cron)

Muchas de las tareas de administración de un servidor se tienen que llevar a cabo de forma periódica. También hay muchas acciones, como la actualización de los logs o las bases de datos internas que utilizan ciertos comandos, que necesitan ejecutarse regularmente para su buen funcionamiento. Por este motivo, es muy importante que el mismo operativo nos proporcione alguna herramienta para poder configurar eficientemente todas estas ejecuciones periódicas.

El daemon cron es el que se encarga de manejar todo el sistema de ejecuciones periódicas. Su organización es muy simple: en el fichero /etc/crontab se guarda la configuración interna del daemon y en los directorios /etc/cron.daily/, /etc/cron.weekly/ y /etc/monthly/, los *shell scripts* de los programas que se ejecutarán diariamente, semanalmente o mensualmente respectivamente. También existe el /etc/cron.d/, donde podemos situar archivos con un formato especial para configurar la ejecución de determinados programas de forma más flexible.

Contenido complementario

Si quisiéramos configurar una consola del sistema para ver todos los logs que se van generando, podríamos añadir la línea "*.* /dev/ttySX"(donde "X" es la consola donde queremos ver los logs) al fichero /etc/syslog.conf y reiniciar el daemon sysklogd

Contenido complementario

Según el servidor que estemos administrando tendremos que tener en cuenta la legalidad vigente (según países) que en algunos casos nos obliga a conservar los ficheros de algún tipo de logs durante un determinado período de tiempo.

ANOTACIONES

Contenido complementario

Muchas de las aplicaciones del sistema necesitan de algún tipo de actualización periódica, generalmente configurada a partir del cron. Si no tenemos el ordenador encendido todo el día, es importante que configuremos adecuadamente el `cron` para que se realicen en algún momento en que sepamos que el ordenador estará encendido.

Contenido complementario

Si instalamos GNU/Linux en un ordenador que no está en funcionamiento todo el día, es recomendable tener instalado el programa `anacron` porque ejecutará los *scripts* configurados con el `cron` adecuadamente (aunque sea en horas diferentes a las previstas).

Contenido complementario

Si utilizamos el fichero `/etc/crontab` para configurar nuestras propias ejecuciones periódicas, cada vez que lo modifiquemos debemos reiniciar el daemon `cron`. Si utilizamos su estructura de directorios, no hace falta.

Generalmente, en el fichero `/etc/crontab` encontramos las siguientes directivas:

```
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:
   /sbin:/bin:/usr/sbin:/usr/bin
#m h dom  mon  dow  user command
25 6 *    *    *    root test -e /usr/sbin/anacron ||
                 run-parts --report /etc/cron.daily
47 6 *    *    7    root test -e /usr/sbin/anacron ||
run-parts --report /etc/cron.weekly
52 6 1    *    *    root test -e /usr/sbin/anacron ||
             run-parts --report /etc/cron.monthly
```

La definición de las variables `SHELL` y `PATH` sirven para indicar al daemon qué intérprete de comandos utilizar y cuál será su `PATH`. Las siguientes tres líneas están formadas por los campos: "`<minuto> <hora> <díaMes> <mes> <diaSetmana> <usuario> <comando>`". Los cinco primeros indican cuándo ejecutar el comando correspondiente (deben ser coherentes) y en el sexto encontramos el usuario que se utilizará para ejecutar el comando especificado en el último. Fijémonos cómo en el fichero de configuración los comandos que se ejecutan una vez al día, una vez a la semana o una vez al mes son los encargados de lanzar los *shell scripts* que se encuentren en los directorios especificados. Si existe el programa `anacron`, se ejecutan con él, si no se utiliza el `run-parts`, que aunque no tiene tantas funcionalidades como el `anacron`, también sirve para poder ejecutar todos los *shell scripts* que se encuentren en un determinado directorio. Esta configuración es la que nos permite toda la estructura de directorios que comentábamos anteriormente. Si quisiéramos, podríamos cambiar las directivas de este archivo para adaptarlas más a nuestras necesidades.

Aunque también podríamos utilizar este mismo fichero de configuración para poner nuestros propios comandos, es más recomendable utilizar la estructura de directorios que nos proporciona el mismo daemon. El único que no aparece en esta configuración es el de `/etc/cron.d/`, que el daemon ya tiene en cuenta automáticamente. En este directorio podemos situar archivos exactamente con la misma sintaxis que en el `/etc/crontab` para programar ejecuciones personalizadas. De esta forma, la flexibilidad es total.

Si para las tareas de administración periódicas del sistema es recomendable utilizar toda esta estructura, cuando son los usuarios los que quieren configurar alguna tarea periódica es más usual utilizar ficheros particulares para cada uno de ellos. Con el comando `crontab` podemos pasar los parámetros "`-u USER -e`" y automáticamente se editará el fichero de configuración particular para el usuario especificado. Los ficheros particulares de los usuarios se guardan en el directorio `/var/spool/cron/crontabs/` (de hecho, el fichero `/etc/crontab` es el mismo que el particular del *root*). Para poder limitar qué usuarios pueden utilizar este daemon, podemos editar los ficheros `/etc/cron.allow` y `/etc/cron.deny`, donde podemos poner, respectivamente, la lista de usuarios a los que permitimos utilizar el cron y a los que no.

Contenido complementario

Utilizando el comando `crontab`, al grabar el fichero se comprueba que la sintaxis sea correcta.

7.4.3. Ejecuciones retardadas (at y batch)

Si bien el `cron` nos permite realizar operaciones cada cierto período de tiempo, el daemon `atd` permite ejecutar un comando o aplicación en un momento determinado. Igual que con el daemon anterior, podemos configurar qué usuarios pueden utilizarlo o no a partir de los ficheros `/etc/at.allow` y `/etc/at.deny`. En este caso, no tenemos fichero de configuración explícito para el daemon, sino que es con el comando `at con el` que podemos especificar en qué momento queremos ejecutar cierta operación con la sintaxis: "`at -f fichero TIEMPO`". Generalmente el fichero suele ser un programa o *shell script* creado por el mismo usuario donde se escriben todas las instrucciones que se quieran ejecutar. La especificación de `TIEMPO` puede llegar a ser muy compleja, pudiendo determinar una `HORA` con el formato "`hh:mm`", un tiempo a partir del momento de ejecución con "`now + XX minutes`", etc. (en su manual se especifican todos los posibles formatos).

Contenido complementario

Todo el sistema de `at` y `batch` funciona con unos mecanismos de cola de ejecución (uno para cada uno de los dos). Aunque podemos configurar más, generalmente sólo con éstas ya tenemos suficiente para realizar cualquier tipo de ejecución retardada en el sistema.

Con `atq` podemos ver qué trabajos tenemos retardados y con `atrm` podemos borrar alguno de los que estén en la cola. Finalmente, si queremos ejecutar todos los trabajos en la cola del `at`, podemos utilizar el comando `atrun`. Este nos permite pasarle el parámetro "`-l LOADAVERAGE`" donde `LOADAVERAGE` debe ser un número que indica a partir de qué momento de carga del sistema se podrán ejecutar los comandos retardados. Esto enlaza directamente con el comando `batch`, que sirve exactamente para lo mismo que el `at` y

Contenido complementario

La **carga del sistema** es un parámetro que nos indica el grado de actividad del ordenador. Con el comando top podemos ver esta carga de forma interactiva.

sigue su misma sintaxis, pero sin necesidad de especificar un tiempo concreto de ejecución. Las operaciones configuradas en esta cola se llevarán a término cuando la carga del sistema baje a menos de 1,5.

De esta forma, cuando necesitamos ejecutar un determinado comando en una hora concreta, deberíamos utilizar el `at`, mientras que para operaciones que queramos realizar sin que entorpezcan el funcionamiento normal del ordenador, deberíamos utilizar el `batch`. En los directorios `/var/spool/cron/atjobs/` y `/var/spool/cron/atspool/` se guardan los ficheros correspondientes a todos estos trabajos retardados.

ANOTACIONES

8. Instalación de aplicaciones

8.1. Introducción

La gestión y manipulación de los paquetes es un aspecto fundamental en cualquier distribución de GNU/Linux. Un paquete es uno o varios programas, librerías o componentes de software empaquetados en un solo archivo preparado para que sea instalado e integrado en el sistema operativo. En el diseño de cualquier distribución es muy importante proporcionar las herramientas necesarias para poder instalar y gestionar adecuadamente estos paquetes. También se deben proporcionar herramientas, especialmente para los desarrolladores de software, para poder crear otros nuevos. En estos paquetes se suelen incluir los ejecutables del programa y sus **dependencias y conflictos** con otras aplicaciones. Las dependencias indican, al instalar un paquete, si necesitan otros programas para que la aplicación funcione correctamente, mientras que los conflictos nos informan de incompatibilidades entre programas instalados y el que queremos instalar. Los sistemas de paquetes están diseñados de esta forma para facilitar la instalación de las nuevas aplicaciones, ya que algunas librerías son utilizadas por más de un programa y no tendría sentido que todas las aplicaciones que las utilizasen las instalaran de nuevo.

Actualmente, la gran mayoría de distribuciones utilizan uno de los dos sistemas de paquetes más extendidos en el mundo del GNU/Linux: los **deb** o los **rpm**. Por un lado, los paquetes deb son los que la distribución de Debian GNU/Linux utiliza en su distribución, mientras que los rpm (*Redhat Package Manager*) son los nativos de RedHat. Las distribuciones basadas en alguna de estas dos generalmente adoptan el sistema de paquetes correspondiente, aunque la mayoría de las otras distribuciones propias también han optado por incorporar alguno de los dos sistemas, ya que actualmente la gran mayoría de programas se empaquetan utilizando estos formatos.

Por otra parte, los programas con licencia GPL o similar también se suelen distribuir con su código fuente (empaquetados y comprimidos

ANOTACIONES

con algún formato estándar, como el tar). A partir de este código fuente, también podemos instalar el programa en nuestro operativo, compilándolo y situando los ejecutables en el lugar donde les corresponda.

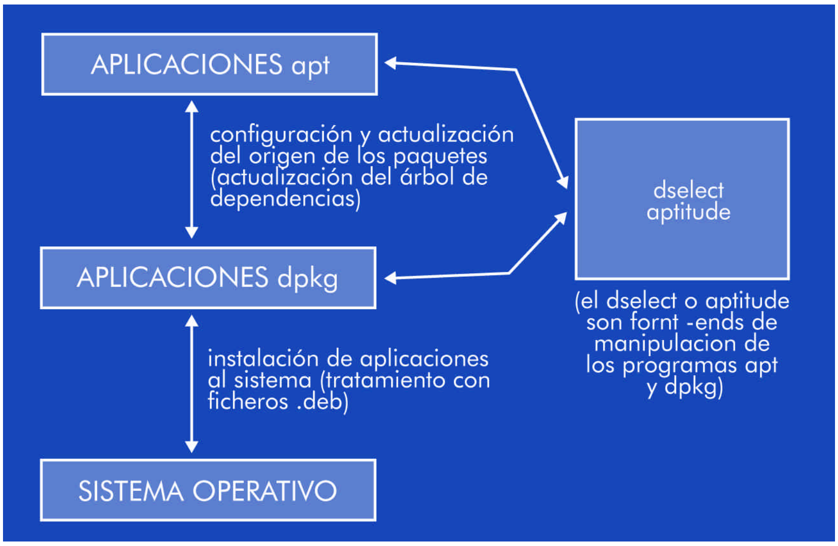
En este capítulo veremos cómo está organizado el sistema de paquetes de la distribución Debian por la gran cantidad de herramientas que se proporcionan y la flexibilidad de su configuración. En la última sección, aprenderemos cómo instalar un programa a partir de su código fuente, ya que en algunos casos podemos encontrarnos que el programa que necesitemos no esté empaquetado. Esta forma de instalación era la que se utilizaba siempre antes de que aparecieran los primeros sistema de paquetes, que surgieron para facilitar todo este proceso.

Contenido complementario

Debian GNU/Linux fue la primera distribución que creó un sistema de paquetes.

8.2. El sistema de paquetes Debian

Las aplicaciones para manipular el sistema de paquetes de Debian GNU/Linux son, básicamente, de dos tipos: los programas `apt` (*Advanced Packaging Tool*) y los `dpkg` (*Debian package*). El conjunto de aplicaciones `apt` sirven para configurar de dónde conseguimos los paquetes, cuáles son los que queremos y resuelven dependencias y conflictos con otros. Los programas `dpkg` sirven para instalar los paquetes, configurarlos, saber cuáles tenemos instalados, etc. Hay otras aplicaciones, como `dselect` o `aptitude`, que sirven para manipular los programas `apt` y `dpkg` proporcionando, en un solo entorno, herramientas interactivas para la manipulación de los mismos. En la siguiente figura podemos ver este esquema:



ANOTACIONES

Los programas que en última instancia se encargan de instalar las aplicaciones son los dpkg. Estos programas descomprimen el fichero ".deb" e instalan el programa. Las aplicaciones apt nos ayudan a localizar las últimas versiones de los programas que necesitamos, copian en el disco los ficheros de las fuentes de donde las hayan extraído (FTP, CD-ROM, etc.) y comprueban dependencias y conflictos de los nuevos paquetes para que se puedan instalar correctamente. Las principales aplicaciones apt son las siguientes:

Contenido complementario

Aunque con las aplicaciones apt también se pueden instalar paquetes, lo único que hacen es llamar a los programas dpkg.

- apt-config: sirve para configurar algunas de las opciones de apt (la arquitectura de nuestro sistema, directorio donde se guardan los archivos, etc.).
- apt-setup: aplicación para configurar las fuentes de los paquetes (de dónde los obtenemos).
- apt-cache: gestión de la caché de paquetes (directorio donde se guardan los archivos ".deb" antes de ser instalados).
- ap-CD-ROM: aplicación para gestionar CD-ROM que contengan paquetes.
- apt-get: actualización, instalación o descarga de los paquetes.

Toda la configuración de apt está en el directorio /etc/apt/. En el fichero /etc/apt/ sources.list es donde se guarda la configuración de las fuentes de los paquetes. Con todas estas fuentes se genera un listado de paquetes disponibles, que podemos consultar e instalar siempre que nos interese. Generalmente, el formato de este archivo sigue la siguiente sintaxis:

```
deb http://site.http.org/debian distribución sección1
                                          sección2 sección3
deb-src http://site.http.org/debian distribución sección1
                                          sección2 sección3
```

El primer campo de cada línea indica el tipo de archivo al que nos referimos: binarios (deb) o código fuente (deb-src). Seguidamente

ANOTACIONES

Contenido complementario

Los sistemas de paquetes también permiten crear paquetes con el código fuente de las aplicaciones. Si sólo nos interesa utilizar la aplicación, no hace falta que descarguemos los paquetes de código fuente.

encontramos la referencia de la fuente de los paquetes, que puede ser un CD-ROM, una dirección de Internet, etc. El campo de distribución indica a `apt` qué versión de Debian GNU/Linux estamos utilizando. Este campo es importante porque cada versión de la distribución tiene sus propios paquetes. En los últimos campos podemos especificar qué tipo de paquetes queremos utilizar.

Si cambiásemos este fichero de forma manual, podríamos utilizar el comando "`apt-get update`" para actualizar todos los paquetes disponibles en el sistema. Para insertar los paquetes de un CD-ROM en el listado de paquetes disponibles, podríamos utilizar "`apt-CD-ROM add`", con lo cual se exploraría el CD insertado y se actualizaría el listado de paquetes del sistema. Si algunas de las fuentes contuvieran paquetes iguales, al instalarlo la misma aplicación `apt` detectaría cual es el más reciente o el que su descarga implica menos tiempo y lo bajaría de la fuente correspondiente. Con el programa `netselect`, además, podríamos configurar más ampliamente todo este sistema de descarga.

Otra opción muy interesante que nos proporciona la mayoría de distribuciones es la de la actualización de paquetes en los que se ha descubierto algún tipo de vulnerabilidad o fallo en su funcionamiento. Con Debian, tan sólo tenemos que añadir la siguiente línea en el archivo `/etc/apt/sources.list`:

```
deb http://security.debian.org/ stable/updates
                              main contrib non-free
```

A medida que se van detectando paquetes críticos, se van poniendo en esta fuente, de forma que con sólo ejecutar "`apt-get update`" se avisa de las nuevas actualizaciones que debemos realizar en el sistema y se reinstalan los paquetes necesarios.

Aunque con los programas `dpkg` podemos manipular cualquier aspecto de los paquetes instalados en el sistema, crear nuevos, modificar los instalados, etc., en este curso sólo repasaremos los más importantes, al nivel de usuario, para que podamos realizar las operaciones básicas con ellos. Los principales programas `dpkg` son los siguientes:

- `dpkg-divert`: nos sirve para manipular el lugar de instalación de algunos de los paquetes instalados en el sistema. Muy útil para evitar algunos problemas de dependencias.

ANOTACIONES

- `dpkg-reconfigure`: con un mismo paquete deb muchas veces se incluye algún mecanismo para configurar algunas de las opciones de la aplicación de forma interactiva. Con esta aplicación podemos volver a configurar el paquete que le indiquemos con los mismos mecanismos utilizados en su instalación.

- `dpkg-scanpackages`: este programa sirve para escanear un determinado directorio del sistema que contenga archivos "`.deb`" para que se genere un archivo de índice. Con este archivo de índice podemos incluir el directorio como una fuente más de `apt`. Muy útil cuando bajamos programas no oficiales de la distribución.

- `dpkg-scansource`: aplicación con las mismas funcionalidades que la anterior pero para paquetes de código fuente.

- `dpkg-split`: programa para dividir y unir un paquete en varios archivos diferentes.

Con estos programas podemos manipular de cualquier forma nuestros paquetes. La aplicación principal, `dpkg`, es la que nos permite instalar, listar o eliminar los paquetes del sistema. Para listar todos los paquetes disponibles le podemos pasar el parámetro "`-l`", con lo cual se mostrará una lista completa de los paquetes y su estado de instalación (instalados, instalados pero no configurados, etc.). Si quisiéramos ver toda la información de un determinado paquete, podríamos utilizar el parámetro "`-p`" seguido del nombre del paquete, con lo cual se muestran todas las dependencias, conflictos con otros paquetes, versión, descripción, etc.

Para instalar nuevos paquetes podemos utilizar el parámetro "`-i`" seguido del nombre del archivo. Si nos da problemas de dependencias, podemos ignorarlas con "`--ignoredepends=X`", donde la "`X`" indica la dependencia, aunque debemos vigilar mucho cómo utilizamos este parámetro porque al ignorar dependencias es posible que el programa instalado no funcione correctamente. Si sólo quisiéramos descomprimir el archivo "`.deb`"para ver qué contiene, también podríamos utilizar "`-x`". Para eliminar los paquetes, debemos pasar "`-r`" seguido del nombre del paquete, que lo elimina del sistema, pero guardando sus archivos de configuración (con "`-P`" se elimina todo).

Contenido complementario

Con `dpkg` también podemos utilizar patrones para seleccionar, instalar, eliminar, . . . los paquetes del sistema.

ANOTACIONES

Otro parámetro muy interesante es el de "--force-things X" (donde la "X" es una de las siguientes opciones), que nos puede ayudar en alguno de los casos que mostramos a continuación:

- "auto-select": selecciona automáticamente los paquetes que se deben instalar o desinstalar con el nuevo paquete que elegimos.
- "downgrade": instala el paquete aunque haya versiones más nuevas del mismo.
- "remove-essential": aunque el paquete esté considerado como esencial en el sistema, lo elimina.
- "depends": no tiene en cuenta las dependencias, las considera como alertas.
- "depends-version": no tiene en cuenta dependencias de versión de los paquetes.
- "conflicts": instala el paquete, aunque entre en conflicto con algún otro del sistema.
- ...:

Aunque todos los programas que hemos ido comentado a lo largo de esta sección tienen muchísimas opciones y existen muchos otros programas, con los que hemos especificado ya nos bastará, con el sistema de paquetes de la distribución que utilicemos, para realizar casi cualquier tarea que sea necesaria. Si bien ya hemos comentado que con programas como el dselect o aptitude ya podremos realizar las tareas básicas de instalación y eliminación de paquetes, es importante conocer adecuadamente estos otros comandos porque para realizar operaciones específicas o para automatizar los procesos de selección e instalación pueden ser muy útiles.

8.3. Compilación de nuevos programas

En la administración de cualquier servidor es muy probable que en algunos casos nos encontremos que debemos utilizar algún pro-

ANOTACIONES

grama que nuestra distribución no tiene o que necesitemos la última versión de un servidor de aplicaciones que todavía no está convenientemente empaquetado, etc. En estos casos, siempre podemos descargarnos el código fuente del programa y compilarlo manualmente. Es importante comprender la diferencia que existe entre compilar un programa para conseguir su ejecutable que descargar directamente el binario. Cuando compilamos un programa, éste utiliza las librerías disponibles en el sistema, mientras que si lo descargamos directamente, lo más probable es que no funcione adecuadamente porque intentará utilizar alguna librería que no será exactamente igual a la que tengamos instalada en el sistema. Por ello, lo más recomendable, cuando necesitemos instalar un nuevo programa del que no disponemos del paquete correspondiente, es compilarlo de nuevo.

Al bajar las fuentes de una aplicación, nos encontraremos con un fichero empaquetado y comprimido con `tar` y `gzip` o similares. Es usual añadir un fichero llamado `README` en el cual se explica paso a paso todas las acciones necesarias para compilar correctamente el programa. Aunque es recomendable leerlo, en la mayoría de casos, el proceso de instalación siempre es el mismo.

Lo primero que debemos hacer para compilar el nuevo programa es descomprimirlo y desempaquetarlo. Una vez hecho esto, dispondremos del código fuente estructurado en varios directorios. En su raíz, podemos encontrar (o no) un fichero llamado `Makefile`. Este archivo indica al compilador qué librerías se utilizan, cómo se deben compilar los archivos de código, etc. Si tenemos este `Makefile`, ya podemos compilar el programa ejecutando `make`. No hace falta que le pasemos ningún parámetro porque por defecto ya busca el fichero de `Makefile` y ejecuta las acciones que se especifican en él. Si el proceso no ha dado ningún error, ya podremos mover el ejecutable generado para ponerlo en alguno de los directorios del `PATH` configurado; de este modo, siempre que lo queramos ejecutar no tendremos que escribir su ruta completa. Muchos `Makefile` proporcionan, asimismo, instrucciones para que podamos ahorrarnos este último paso. Generalmente, ejecutando "`make install`" el mismo programa se encarga de situar adecuadamente los binarios y, si existieran, los archivos de documentación. Finalmente, si no nos interesara guardar el código fuente del

Contenido complementario

Al compilar un programa es muy probable que necesitemos tener instaladas en el sistema las fuentes o las cabeceras de las librerías que utiliza. Generalmente, estos paquetes suelen tener el mismo nombre que la librería, pero añadiendo "`-dev`" (de desarrollo) al final.

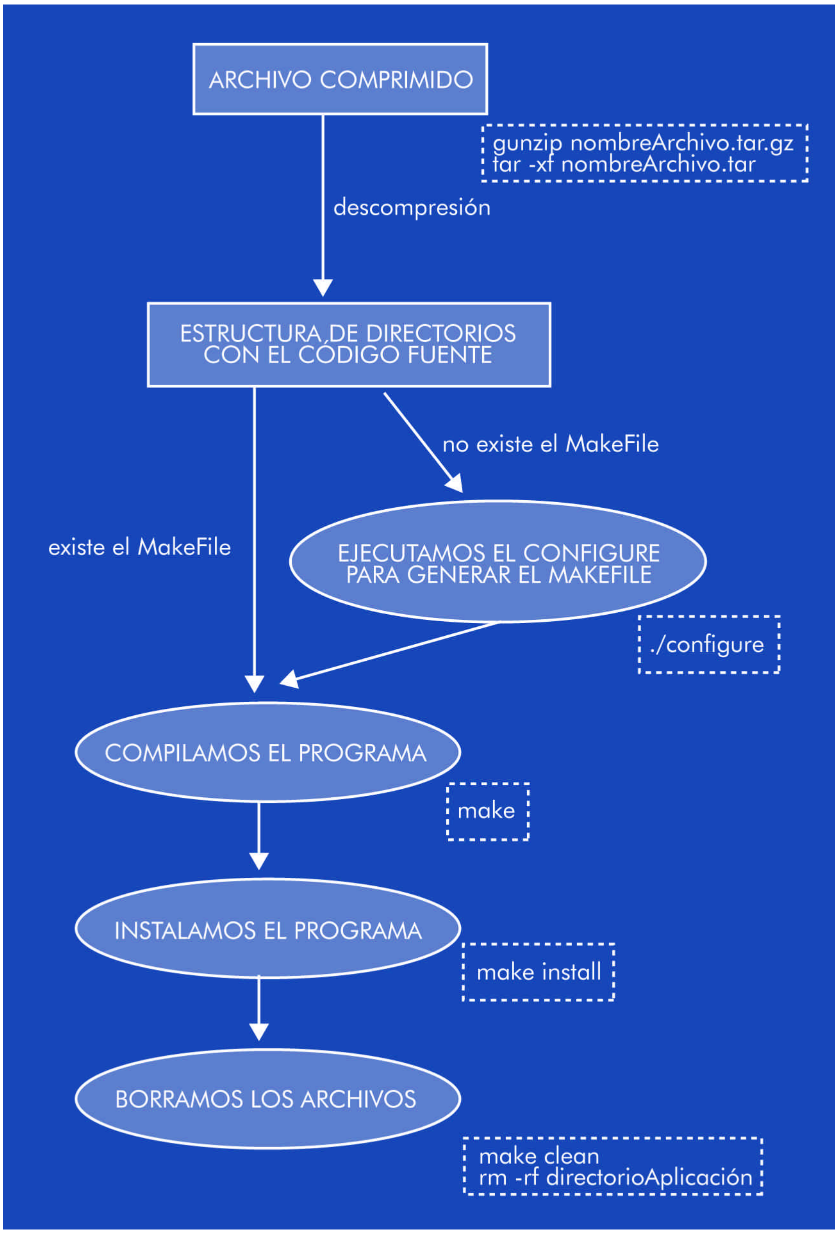
Contenido complementario

El proceso de compilación de un programa puede durar desde segundos a horas, según la aplicación. Por ejemplo, la compilación del núcleo Linux va desde 5 o 10 minutos a las 2 horas (según la versión del núcleo y la potencia del ordenador).

ANOTACIONES

programa, ya podemos eliminar todo el contenido de los directorios creados.

Si el programa no incorpora el archivo de `Makefile`, generalmente se suele incluir algún *shell script* para generar automáticamente este fichero (habitualmente, este *script* se suele nombrar `configure`). Al ejecutar este *shell script* se comprobará que el sistema tenga instaladas todas las librerías necesarias para una buena compilación y, si faltara alguna, se daría un mensaje de aviso. Una vez ejecutado correctamente este *shell script,* ya dispondremos del `Makefile`, con lo que el proceso vuelve a ser el mismo que anteriormente. En la siguiente figura podemos ver todo esto de forma gráfica:



Aunque la mayoría del código fuente de los programas se organiza de la forma como hemos expuesto, también es posible que nos encontremos con otros tipos de instalación. Hay alguno que incorpora menús, entornos en X u otros métodos más amenos. Últimamente, también empiezan a aparecer algunas aplicaciones que permiten realizar todo el proceso de instalación bajando el código directamente de Internet.

ANOTACIONES

9. Taller de configuraciones básicas

9.1. Introducción

En el segundo taller hemos aprendido a instalar un sistema básico para que, a partir de éste, podamos empezar a montar un sistema a medida de nuestras necesidades. Éste será el objeto de estos dos últimos talleres. En el primero, después de concretar algunos aspectos referentes a la configuración del sistema de instalación de paquetes, quedarán sentadas las bases para poder instalar las aplicaciones necesarias. A continuación, aprenderemos a utilizar las distintas herramientas que nos ofrece Debian para la gestión de paquetes, es decir, para instalar, desinstalar, actualizar, etc. aplicaciones, e instalaremos y configuraremos algunas de ellas, que son comunes y necesarias en la mayoría de sistemas tipo UNIX. El último taller está plenamente orientado al sistema gráfico. En él aprenderemos a instalarlo, configurarlo, adaptarlo a nuestras preferencias y, por último, aprenderemos a instalar y utilizar aplicaciones que se sirven de este sistema para poder correr.

9.2. El gestor de arranque

En primer lugar debemos asegurarnos de instalar un sistema de arranque que sea capaz de gestionar sin problemas los posibles sistemas operativos que tengamos instalados en nuestro ordenador. Esta cuestión ya se abordó superficialmente en el taller anterior, durante el proceso de instalación. Si ya tenemos instalado el gestor de arranque y hemos comprobado su correcto funcionamiento, podemos pasar a la sección siguiente; pero si no es así, es decir, si para arrancar nuestro sistema operativo necesitamos del disquete de rescate que creamos durante el proceso de instalación, ha llegado el momento de instalar un sistema de gestión de arranque en el disco duro para evitar la tediosa tarea de utilizar cada vez el disquete. De cualquier modo, siempre es interesante tener instalado un sistema gestor de arranque, ya que nos permitirá, entre otras cosas, poder arrancar diversos *kernels*.

ANOTACIONES

Tal como se ha expuesto, hay distintos gestores de arranque, entre ellos Lilo y Grub. Grub es un poderoso sistema gestor de arranque, del proyecto GNU, que se caracteriza por poder gestionar correctamente el arranque de cualquier sistema operativo que tengamos instalado en nuestro ordenador, no obstante, su uso y su configuración son un tanto complejos. Lilo es el gestor que se diseñó inicialmente para gestionar los arranques de los *kernels* Linux, se caracteriza por tener un sistema de instalación más intuitivo, además de ofrecer la posibilidad de forma sencilla, mediante una sola línea de comandos, de reestablecer la imagen de la MBR anterior si algo ha ido mal; aspecto muy interesante, sobre todo si en nuestro disco duro conviven distintos sistemas operativos, ya que escribir en la MBR puede inutilizar alguno de ellos, en especial si se trata de sistemas Microsoft™.

De cualquier forma, si no disponemos de gestor de arranque y queremos instalar uno, lo primero que debemos hacer es arrancar el sistema operativo que tenemos y comprobaremos si la aplicación está instalada, haciendo, por ejemplo, un man de la aplicación.

9.2.1. Instalación de Lilo

El archivo de configuración de Lilo se encuentra en `/etc/lilo.conf`. El archivo está bien documentado y, por consiguiente, es fácil de modificar mediante un editor de texto y adaptarlo a nuestras necesidades. Una vez hayamos hecho las modificaciones que consideramos oportunas, lo que haremos es ejecutar como *root* la instrucción `lilo` para transferir a la MBR el contenido del fichero de configuración.

Como medida de precaución, lo que haremos inmediatamente después es reiniciar el ordenador, por ejemplo mediante el comando `reboot`, y comprobaremos que todos los sistemas operativos que tenemos instalados arrancan correctamente. Si no fuese así, lo que se debe hacer es volver a arrancar nuestro recién instalado sistema operativo (mediante Lilo, si éste es capaz de hacerlo, o mediante el disquete de rescate –el que hemos utilizado hasta el momento para arrancar el sistema-) y ejecutar la instrucción siguiente para reestablecer la copia de seguridad del la MRB que Lilo ha hecho antes de

ANOTACIONES

escribir sobre ella; con ello estaremos en la misma situación que estábamos antes de ejecutar `lilo`:

```
brau:~# lilo -U
```

Una vez hecho esto, debemos proceder a estudiar el porqué de los errores y tratar de corregirlos; ha llegado el momento de empezar a leer documentación, empezando por los mans, las FAQ, la documentación que podemos encontrar en /usr/share/doc/lilo/ etc. Una vez hechos los cambios pertinentes, repetiremos el proceso anterior.

Puede suceder que tras diversos intentos lleguemos a la conclusión de que: o Lilo no puede o no sabemos configurarlo correctamente para arrancar todos los sistemas operativos que tenemos instalados en el ordenador. Ha llegado el momento de probar con otros gestores, como puede ser Grub.

9.2.2. Instalación de Grub

Grub no viene instalado por defecto, por este motivo, antes de usarlo debemos proceder a su instalación. Para ello, podemos proceder a leer las secciones posteriores para informarnos sobre el sistema gestor de paquetes, su configuración y su uso; o simplemente, en caso de que durante la instalación hayamos introducido en la base de datos los contenidos de todos los CD o que hayamos optado por una instalación por red, ejecutaremos la línea siguiente:

```
brau:~# apt-get install grub grub-doc
```

Si no deseamos instalar el paquete de documentación, se puede omitir el último parámetro de la línea anterior.

Una forma recomendable de trabajar con Grub es hacer, antes de escribir en la MRB, las pruebas que se estimen oportunas sobre un disquete y probar su correcto funcionamiento arrancando desde éste. Para ello, lo primero que haremos es copiar los ficheros necesarios en /boot/:

```
brau:~# cp /usr/lib/grub/i386-pc/stage* /boot/
brau:~# cp /usr/share/doc/grub/examples/menu.lst /boot/
```

ANOTACIONES

Una vez hecho esto, editaremos el fichero de configuración de Grub, /boot/menu.lst, y lo adaptaremos a nuestras necesidades. Esta operación puede ser un tanto compleja, debido, entre otras cosas, al cambio de nomenclatura en lo que hace referencia a los discos duros; así pues, para identificar en qué dispositivo se halla un fichero en concreto, podemos ayudarnos del comando `find` una vez hayamos entrado en el modo de comandos de Grub mediante `Grub`. Así pues, por ejemplo, para localizar un fichero, cuya ubicación seguro que necesitaremos especificar en el fichero de configuración de Grub, como puede ser `/vmlinuz`, procederemos de la manera siguiente:

```
brau:˜# grub

   GRUB version 0.91 (640K lower / 3072K upper memory)

  [ Minimal BASH-like line editing is supported. For the
   first
word, TAB lists possible command completions.
   Anywhere else TAB lists the possible completions
   of a device/filename. ]

grub> find /vmlinuz
  (hd2,0)
```

Grub cuenta con muchos más comandos, para ver un listado de algunos de ellos basta con pulsar la tecla TAB en la línea de comandos para obtener lo siguiente:

```
grub>
   Possible commands are: blocklist boot cat chainloader
cmp color configfile deb ug device displayapm displaymem
embed find fstest geometry halt help hide impsp robe initrd
install ioprobe kernel lock makeactive map md5crypt module
moduleno unzip partnew parttype password pause quit read
reboot root rootnoverify savede fault serial setkey setup
terminal testload testvbe unhide uppermem vbeprobe

grub>
```

Para obtener ayuda de algún comando en concreto, basta con teclear `help` seguido del nombre de dicho comando. Una buena forma de trabajar sobre nuestro fichero de configuración de Grub, /

ANOTACIONES

boot/menu.lst es abriendo una nueva sesión en una tty distinta e ir modificando dicho fichero a medida que vayamos trabajando sobre la interfaz de comandos de Grub.

Una vez hayamos terminado de editar el fichero de configuración, y tras introducir un disquete virgen en la disquetera, teclearemos –sustituyendo el carácter X por el número de disco duro, e Y por la partición correspondiente– la línea de comandos siguiente:

```
grub> install (hdX,Y)/boot/stage1 d (fd0) (hdX,Y)/boot/stage2 p (hdX,Y)/boot/menu.lst
```

Con la línea anterior hemos transferido al disquete (fd0) la información de la MBR (stage1), Grub y su interfaz de comandos (stage2), y el menú de arranque que hayamos configurado en el fichero /boot/menu.lst. Estamos, pues, en condiciones de reiniciar la máquina, arrancar mediante el disquete que acabamos de crear y comprobar si nuestra configuración es correcta.

Es muy interesante disponer de un disquete con Grub, ya que, mediante su interfaz de comandos, podemos intentar arrancar directamente los distintos sistemas operativos que tengamos instalados, con el objeto de ir haciendo pruebas para depurar el contenido del fichero /boot/menu.lst. A modo de ejemplo, se expone el procedimiento que hay que seguir para arrancar un sistema Microsoft™ instalado como (hd0,0) y a continuación los comandos necesarios para arrancar un sistema GNU/Linux instalado como (hd1,0):

Para un sistema Microsoft:

```
grub> rootnoverify (hd0,0)
grub> makeactive
grub> chainloader +1
grub> boot
```

Para un sistema GNU/Linux:

```
grub> kernel (hd1,0)/vmlinuz root=/dev/hdc1
grub> boot
```

ANOTACIONES

Una vez tengamos en nuestro disquete el sistema de arranque definitivo que deseamos implementar, simplemente debemos transferir esta misma configuración a la MBR del disco duro de arranque, desde la misma línea de comandos del disquete teclearemos:

```
grub> install (hdX,Y)/boot/stage1 d (hd0,0) (hdX,Y)/boot/stage2 p (hdX,Y)/boot/menu.lst
```

Como se ha dicho, utilizar Grub es un tanto más complejo que Lilo, por esta razón se recomienda que antes de embarcarse en su instalación y posterior configuración, se lea detenidamente los mans y la documentación que se ha instalado con el paquete `Grub-doc`, también accesible en `http://www.gnu.org/software/grub/`.

9.3. El sistema de paquetes

Ha llegado el momento de analizar y aprender a utilizar el sistema de paquetes de Debian, probablemente el más sólido y fiable de cuantos existan en el mundo GNU.

En las subsecciones siguientes aprenderemos a configurar su base de datos, a manipularla, a instalar paquetes, actualizarlos, etc. En muchas ocasiones hay distintas maneras de obtener el mismo resultado. Se expondrán algunas de ellas, con dos objetivos principales: el primero, para que el lector pueda optar por la que más le interese, y el segundo, porque es interesante conocer siempre más de una solución a un mismo problema, por si acaso alguna de ellas fallara.

Actividad

13. Para la plena comprensión del funcionamiento del sistema de paquetes Debian, se recomienda la lectura de:

- APT HOWTO: http://www.de.debian.org/doc/manuals/apt-howto/index.en.html

ANOTACIONES

- http://www.de.debian.org/doc/manuals/debian-faq/ch-pkg basics.en.html
- http://www.de.debian.org/doc/manuals/debian-faq/ch-pkgtools.en.html
- Los mans de: `apt`, `apt-cache`, `apt-get`, `sources.list`, `dpkg` y `dselect`.

Nota: Para hacer estas lecturas más asequibles, podemos esperar a realizarlas cuando tengamos configurada la impresora y hayamos aprendido a imprimir mans.

9.3.1. /etc/apt/sources.list

El archivo `/etc/apt/sources.list` es el corazón de la configuración del sistema de paquetes de Debian. Al tratarse de un fichero de texto, como la mayoría de ficheros de configuración en los sistemas UNIX, lo podemos editar manualmente, o bien mediante algunas herramientas de las que dispone el sistema para tal fin.

El contenido de este fichero dependerá en gran medida de la velocidad con que podamos acceder a Internet, si es que lo podemos hacer. Pero en ningún caso debemos olvidar ejecutar la instrucción siguiente como *root*, una vez hayamos modificado el fichero:

```
brau:/etc/apt# apt-get update
```

Si no lo hicieramos, la base de datos del sistema de paquetes no se actualizaría y, en consecuencia, ninguno de los cambios realizados surgiría efecto. En una instalación donde los paquetes se obtengan de forma remota, este comando debe ser ejecutado periódicamente para ir actualizando la base de datos; por este motivo, no es mala idea incluirlo dentro del sistema cron.

Cada línea de este fichero hace referencia a una fuente de paquetes y los campos van separados por un espacio. En primer lugar especificaremos si la fuente de paquetes es de paquetes binarios `deb` o si

ANOTACIONES

es de código fuente `deb-src`. En el segundo campo especificaremos la forma de acceder a éstos: `CD-ROM`, `http`, `ftp`, etc. seguido de la dirección de acceso. Los campos restantes hacen referencia al tipo de paquetes al que queremos acceder por esta línea.

/etc/apt/sources.list con acceso rápido a Internet

En el mejor de los casos dispondremos de un acceso rápido a Internet. Esto, probablemente, ya nos habrá permitido hacer la instalación del sistema básico por red, además de disponer siempre de las últimas versiones de los paquetes. Se trata, además, de la forma más cómoda de trabajar con paquetes, ya que no tenemos ni siquiera que preocuparnos de insertar el CD correspondiente para hacer una instalación.

De lo que debemos cerciornarnos, antes que nada, es de que el contenido de `/etc/apt/sources.list` sea correcto. A continuación se muestra un ejemplo:

```
deb http://ftp2.de.debian.org/debian/ stable main non-free contrib
deb-src http://ftp2.de.debian.org/debian/ stable main non-free contrib
deb http://non-us.debian.org/debian-non-US stable/non-US main contrib non-free
deb-src http://non-us.debian.org/debian-non-US stable/non-US main contrib non-free
deb http://security.debian.org/ stable/updates main contrib non-free
```

Todas las direcciones anteriores son oficiales, es decir, reconocidas por Debian. Aparte de estas fuentes, también se pueden utilizar paquetes no oficiales, que por no ser oficiales no significa que carezcan de calidad suficiente como para ser incluidos en nuestro fichero de configuración. Una buena dirección para obtener información acerca de la localización de estos paquetes es `http://www.apt-get.org`.

Una vez editado el fichero y salvado, ejecutaremos el comando "`apt-get update`" y tras unos instantes, en los cuales el sistema de paquetes reconfigurarará su base de datos y visualizará por pantalla los distintos accesos que se realizan, ya tendremos acceso a los nuevos paquetes.

ANOTACIONES

`/etc/apt/sources.list` sin acceso rápido a Internet

En el caso de no disponer de conexión a Internet o de ser lenta, debemos optar, sin dudarlo, por utilizar el juego de CD de la distribución para ir instalando los distintos paquetes. Si durante el proceso de intalación no hemos insertado todos los CD, habrá llegado el momento de hacerlo. Insertamos el primer CD-ROM en el lector y usamos el comando `apt-CD-ROM` para incluir sus contenidos en la base de datos:

```
brau:/etc/apt# apt-CD-ROM add
Using CD-ROM mount point /CD-ROM/
.
.
.
Repeat this process for the rest of the CDs in your
set.
```

Llegados a este punto, repetiremos el mismo proceso para todos y cada uno de nuestros CD de la distribución. Asimismo, puede utilizarse el mismo procedimiento para incorporar datos procedentes de CD-ROM no oficiales.

Una vez tengamos configurado nuestro acceso a Internet, si lo estimamos oportuno, podemos incluir fuentes de paquetes de acceso reomoto. Para ello editaremos el fichero `/etc/apt/sources.list` y despues de ejecutar "`apt-get update`", tendremos disponibles los nuevos paquetes.

9.3.2. apt

apt es acrónimo de *Advanced Package Tool*, que, como ya se ha dicho en diversas ocasiones, es el sistema básico encargado de la administración de paquetes de las distribuciones basadas en Debian. apt pone a nuestra disposición esencialmente dos herramientas: `atp-get` y `apt-cache`. El primer comando lo puede utilizar única y exclusivamente el *root* del sistema, ya que es la herramienta de gestión de paquetes: instalación, desinstalación, actualización, etc., mientras que el segundo, al ser un comando orientado a la búsque-

ANOTACIONES

da de información dentro de la base de datos, ya sean paquetes instalados o sin instalar, puede ser utilizado por cualquier usuario.

Con el objeto de facilitar el manejo de paquetes, se han desarrollado otras aplicaciones que corren por encima de apt, como puede ser el middle-end `dpkg` o los front-end `dselect` o `aptitude`. Pero antes de adentrarnos en los distintos sistemas de administración de paquetes, debemos conocer algunos conceptos acerca de éstos y de su relación con el sistema y el sistema de gestión.

Tipos de paquetes según su prioridad

Dentro del sistema de paquetes se distinguen cinco tipos distintos según su grado de dependencia con el mismo sistema. Por orden decreciente de prioridad se clasifican como:

Required. Se trata de paquetes indispensables para el correcto funcionamiento del propio sistema.

Important. Se trata de paquetes que deberían estar presentes en cualquier sistema tipo UNIX.

Standard. Se trata de paquetes que comúnmente se encuentran en un sistema GNU/Linux. Por lo general son aplicaciones de tamaño reducido, pero que ya no son indispensables para el sistema.

Optional. Se trata de paquetes que pueden estar o no, presentes en un sistema GNU/Linux. Entre otros, dentro de este grupo se hallan todos los paquetes referentes al sistema gráfico, ya que éste no se considera indispensable. En realidad, en muchos servidores, con el objeto de aumentar su rendimiento se prescinde del entorno gráfico.

Extra. Son paquetes que, o bien presentan conflictos con paquetes con prioridad superior a la suya o bien porque requieren de configuraciones especiales que no los hacen aptos para ser integrados como *optional*.

ANOTACIONES

Podemos determinar a qué grupo pertenece un paquete en concreto mediante, por ejemplo, la sentencia "`apt-cache show <nombre del paquete>`" y consultar el contenido del campo `Priority`:

Grado de dependencia entre paquetes

apt se caracteriza por su gran consistencia a la hora de gestionar las dependencias que existen entre paquetes. Puede, por ejemplo, que una determinada aplicación que queremos instalar dependa de una librería y, en consecuencia, de otro paquete que no tengamos instalado. En este caso, apt nos informará de esta dependencia y nos preguntará si queremos que, junto con la aplicación que vamos a instalar, se instale el paquete que contiene dicha librería. Las relaciones entre paquetes se clasifican de la manera siguiente:

depends. El paquete que queremos instalar depende de estos paquetes y, por consiguiente, si queremos que este paquete funcione correctamente, debemos permitir que apt instale el resto de ellos.

recommends. El responsable del mantenimiento del paquete ha estimado que normalmente los usuarios que vayan a instalar este paquete también usarán los que recomienda.

suggests. Los paquetes que se sugieren permiten obtener un mayor rendimiento del paquete que vamos a instalar.

conflicts. El paquete que vamos a instalar no funcionará correctamente si estos otros paquetes están presentes en el sistema.

replaces. La instalación de este paquete implica la desinstalación de otros paquetes.

provides. El paquete que vamos a instalar incorpora todo el contenido de los paquetes mencionados.

Podemos determinar las dependencias de un paquete, por ejemplo, la sentencia "`apt-cache depends <nombre del paquete>`".

ANOTACIONES

Acciones sobre los paquetes

Mediante los flags siguientes, dpkg o dselect nos informará acerca de lo que el usuario pretende hacer con dichos paquetes:

unknown. Nunca se ha hecho referencia a dicho paquete.

install. Se quiere instalar o actualizar el paquete.

remove. Se quiere desinstalar el paquete, pero manteniendo sus ficheros de configuración (comúnmente situados en `/etc/`).

purge. Se quiere desinstalar por completo el paquete, inclusive sus ficheros de configuración.

hold. Se quiere indicar que no se quiere realizar ninguna operación sobre este paquete (es decir, que se mantenga hasta nuevo aviso su versión y su configuración).

Estado de instalación de los paquetes

Dentro del sistema un paquete se puede hallar:

installed. El paquete ha sido instalado y configurado correctamente.

half-installed. La instalación del paquete se ha comenzado, pero, por alguna razón, no ha terminado.

not-installed. El paquete no está instalado en el sistema.

unpacked. El paquete ha sido desempaquetado, pero no configurado.

half-installed. El paquete ha sido desempaquetado y se ha iniciado el proceso de configuración, pero por alguna razón éste no ha terminado.

config-files. Sólo existen, en el sistema, los archivos de configuración de dicho paquete.

ANOTACIONES

apt-cache

Como ya se ha dicho, apt-cache es un comando orientado al análisis del sistema de paquetes y, por tanto, al no ser un arma potencialmente peligrosa para el sistema, es accesible a todos sus usuarios. Los parámetros más utilizados para este comando son los siguientes:

search pattern. Busca en la base de datos los paquetes cuyo nombre contenga *pattern* o en cuya descripción aparezca *pattern* (si el resultado es un lista extensa debido a que *pattern* es muy general, se pueden utilizar *pipes* y grep para filtrar estos resultados).

show package. Innforma acerca del paquete.

policy package. Informa acerca del estado de instalación, la versión y revisión del paquete, y su procedencia.

depends package. Explicita las dependencias del paquete.

show package. Muestra las dependencias directas y las reversas del paquete.

apt-get

apt-get es el comando que se utiliza para gestionar los paquetes del sistema. Por este motivo su uso está restringido al *root* del sistema. Los parámetros más utilizados para este comando son los siguientes:

install package. Instala el paquete. Si éste depende de paquetes que no se encuentran en el sistema, apt nos informará de ello, y nos preguntará si junto con el paquete en cuestión queremos instalar los paquetes de los que depende y que no están instalados; por lo general es interesante seguir los consejos de apt.

update. Actualiza la base de datos de apt. Este comando debe ejecutarse cada vez que se modifica el archivo /etc/apt/sources.list.

upgrade. Fuerza la actualización de todos los paquetes instalados en el sistema a la última versión disponible.

ANOTACIONES

remove ***package***. Elimina el paquete, sin eliminar los ficheros de configuración, de cara a posibles reinstalaciones.

remove -purge ***package***. Elimina por completo el paquete, incluyendo sus archivos de configuración.

clean. Elimina las copias caducadas de los paquetes que se ha ido instalando, proceso en el cual se almacena de manera automática una copia del paquete sin desempaquetar en `/var/cache/apt/archives` cuando se instala un paquete. Comando muy útil de cara a liberar espacio del disco duro, ocupado por ficheros que, probablemente, nunca más serán utilizados.

autoclean. Elimina todas las copias no desempaquetadas de los paquetes, independientemente de su vigencia.

9.3.3. dpkg

dpkg es acrónimo de *Debian Package Manager* y fue concebido como *back-end* de apt. Los parámetros más utilizados son los siguientes:

- **l.** Para listar todos los paquetes de la base de datos y su estado de instalación (generalmente esta opción se combina con `grep`).
- **L** ***package***. Para listar los ficheros contenidos en el paquete.
- **r** ***package***. Tiene el mismo efecto que `apt-get remove` *package*.
- **P** ***package***. Tiene el mismo efecto que `apt-get remove --purge` *package*.
- **p** ***package***. Tiene el mismo efecto que `apt-get show` *package*.
- **s** ***package***. Describe el estado de instalación del paquete.
- **S** ***file***. Busca a qué paquetes pertenece el fichero.

ANOTACIONES

9.3.4. `dselect`

dselect es una GUI (*Graphical User Interface*) que corre sobre apt. Para entrar en ella, basta con teclear el comando `dselect`, y mediante los menús de esta interfaz ir seleccionando los distintos paquetes sobre los cuales queremos operar y especificar qué tipo de operación deseamos hacer sobre ellos.

9.3.5. `aptitude`

aptitude es otra GUI que corre sobre apt. Por defecto no viene instalada, por lo que hay que hacerlo antes de proceder a su uso:

```
brau:/etc/apt# apt-get install aptitude
```

Una vez instalada, la lanzamos mediante el comando `aptitude`, y enseguida veremos que su uso es igual o más simple que el de dselect, ya que dispone de menús desplegables accesibles mediante F10.

9.4. locales: configuración regional

Aunque aparentemente nuestro teclado funcione correctamente, ya que podemos utilizar acentos, diéresis y otros caracteres no ingleses, a medida que vayamos adaptando el sistema a nuestras necesidades, y especialmente cuando instalemos el sistema gráfico y hagamos correr aplicaciones sobre él en el próximo taller, nos daremos cuenta de que esto no es así. Podemos, pues, en este punto proceder a configurar correctamente estos aspectos para no tenerlo que hacer más adelante.

En primer lugar comprobaremos si el paquete `locales` está instalado:

```
brau:/# dpkg -l| grep locales
ii locales       2.2.5-11.2       GNU C Library: National Language (locale) da
```

Si no obtenemos la última línea, debemos proceder a instalar el paquete y configurarlo:

```
brau:/# apt-get install locales
```

Y si ya disponemos de él, teclearemos la siguiente línea para reconfigurarlo:

```
brau:/# dpkg-reconfigure locales
```

De las muchas opciones que se nos ofrecen, elegimos `[*] es ES ISO-8859-1`, es decir, nos situamos sobre dicha opción y pulsamos la barra espaciadora. Mediante TAB, nos situamos sobre `OK` y pulsamos INTRO. En la próxima pantalla seleccionamos `C`.

De vuelta a la línea de comandos, editamos el fichero `/etc/environment` para dejarlo de la manera siguiente:

```
LC_ALL=es_ES
LANGUAGE=en_US
LC_TYPE=es_ES
LC_MESSAGES=ISO8859-1

LANG=C
```

Para hacer efectivo el cambio, basta con teclear el comando `locale-gen`, y saldremos de todas las sesiones que tengamos abiertas para cargar la nueva configuración.

Lectura complementaria

Para saber más acerca de locales, se recomienda visitar la página:
http://www.uniulm.de/ s smasch/locale/.

9.5. Configuración de *man* y su *pager*

Puede que al intentar invocar `man` nos dé un error del tipo:

```
Reformatting man(1), please wait...
sh: /usr/bin/pager: No such file or directory
sh: exec: /usr/bin/pager: cannot execute: No such file or directory
man: command exited with status 32256: /usr/bin/zsoelim /tmp/zmanZpjojO |
    /usr/bin/tbl | /usr/bin/nroff -mandoc -Tlatin1 | exec /usr/bin/pager -s
```

ANOTACIONES

En este caso, no tenemos ningún paginador asignado a `/usr/bin/pager`, de modo que `man` lo pueda utilizar para mostrarnos los contenidos de ayuda. Es más, probablemente no tengamos ningún paginador instalado. En ambos casos, lo que haremos es intentar instalar `less`, seguramente el paginador más utilizado para `man`:

```
brau:/# apt-get install less
```

En el caso de que el paquete no esté instalado, se nos abrirá una pantalla de configuración, a la que en principio, contestaremos negativamente, opción por defecto. Con esto ya habremos instalado `less`, y probablemente ya se haya asignado él mismo como paginador de `man`. Si no fuera así (lo podríamos probar ejecutando un `man man`, por ejemplo), o si `less` ya estuviera instalado, utilizaríamos el comando siguiente para asignarlo como paginador (también se puede utilizar para cambiar de paginador, por ejemplo, para cambiar a `more`, `jless`, o cualquier otro paginador):

```
brau:/# update-alternatives --config pager
```

Lo que hace el comando anterior, en el caso del paginador, y en general con el resto de aspectos configurables mediante él, es crear enlaces simbólicos. Para el caso del paginador éstos son:

```
/etc/alternatives/pager -> /usr/bin/less
/usr/bin/pager -> /etc/alternatives/pager
```

9.6. El archivo principal de arranque, `/etc/inittab`

Aunque el proceso de arranque de un sistema GNU/Linux es complejo, en esta sección sólo se pretende trabajar sobre uno de los ficheros principales de este proceso: `/etc/inittab`. Este archivo indica al proceso de arranque, entre otros, a qué *runlevel* se entrará, y definirá qué procesos se arrancarán de forma automática durante el proceso de arranque. Para saber en qué *runlevel* nos hallamos, basta con teclear el comando `runlevel`. Para cambiar de *runlevel*, como *root*, usaremos la instrucción `init <runlevel de destino>`.

ANOTACIONES

Es interesante abrir este fichero e irse familiarizando con su contenido, ya que esto nos permitirá comprender mejor el proceso de arranque de un sistema GNU/Linux.

9.7. Montaje de dispositivos, /etc/fstab

/etc/fstab es el fichero que contiene la información acerca de las particiones y dispositivos que se montarán de forma automática durante el proceso de arranque, y las que se pueden montar posteriormente, así como también establece quién puede hacerlo. A continuación, se muestra el posible contenido de este fichero y se pasa a analizarlo:

```
# /etc/fstab: static file system information.
#
# <file system>  <mount point>   <type>    <options>           <dump>
<pass>
/dev/hdg1        /               ext3      errors=remount-ro   0        1
/dev/hdg2        none            swap      sw                  0        0
proc             /proc           proc      defaults            0        0
/dev/fd0         /floppy         auto      user,noauto         0        0
/dev/hdg5        /usr            ext3      defaults            0        2
/dev/hdg6        /var            ext3      defaults            0        2
/dev/hdg7        /home           ext3      defaults            0        2

/dev/CD-ROM /CD-ROM iso9660 ro,user,noauto 0 0

/dev/hdc1 /mnt/hdc1 ntfs ro,user,noauto,gid=windows,umask=0007,utf8 0 0
/dev/hde1 /mnt/hde1 ntfs ro,user,noauto,gid=windows,umask=0007,utf8 0 0
/dev/hde5 /mnt/hde5 ntfs ro,user,noauto,gid=windows,umask=0007,utf8 0 0

/dev/hde6 /mnt/hde6 vfat utf8,user,noauto 0 0
/dev/hde7 /mnt/hde7 vfat utf8,user,noauto 0 0
```

ANOTACIONES

Las primeras líneas las ha generado automáticamente el proceso de instalación y en ellas podemos ver cómo están distribuidos los distintos directorios dentro de la estructura puramente GNU/Linux. Quizás la línea que más llama la atención sea la proc /proc proc defaults 0 0; ésta es la encargada del montaje del directorio virtual proc, del cual ya se habló en el primer taller.

Más interesantes son las líneas tipo `/dev/hdc1 /mnt/hdc1 ntfs utf8,ro,noauto,user,gid=windows,umask=0007,utf8 0 0`. En ellas se especifica el punto de origen y el punto de montaje de particiones pertenecientes al sistema operativo Windows2000™, es decir, de tipo ntfs. En estas particiones no se puede escribir desde GNU/Linux, aunque sí que se puede leer su contenido, esto se ve reflejado en las opciones `ro,noauto,user,gid=windows,umask=0007,utf8` (es fundamental no dejar ningún espacio en blanco entre opciones, ya que este carácter es el que se utiliza para separar los campos en este fichero). La primera indica que se trata de una partición de sólo lectura; la segunda, que no se monte automáticamente durante el proceso de arranque del sistema; la tercera indica que esta partición la puede montar cualquier usuario; la cuarta opción indica que sólo podrán acceder a ella los miembros pertenecientes al grupo Windows (definido en el fichero `/etc/group`); la penúltima opción establece la antimáscara de montaje y la última, la tabla de códigos a utilizar.

Hay que tener presente que por defecto el *kernel* que hemos instalado no soporta el tipo ntfs. Por tanto, hay que cargar el módulo correspondiente mediante el comando `modconf` y seleccionar la `opcion kernel/fs/ntfs`.

Las últimas líneas del fichero anterior van destinadas a montar particiones fat32, sobre las cuales sí que es posible escribir desde GNU/Linux. Por esta razón es buena idea disponer siempre de una pequeña partición con este tipo de formato, ya que será accesible tanto desde GNU/Linux, como desde Windows™.

Si bien es cierto que es posible montar un sistema de ficheros desde la línea de comandos, como por ejemplo se haría para montar el CD-ROM,

```
brau:/etc/apt# mount /dev/CD-ROM /CD-ROM -t iso9660 ro
```

es mucho más cómodo tener la información introducida en el archivo `/etc/fstab`, ya que esto nos permitirá hacer lo mismo tecleando tan sólo:

```
brau:/etc/apt# mount /CD-ROM
```

ANOTACIONES

9.8. Configuración de dispositivos

Una vez sentadas las bases para la administración de paquetes, podemos abordar la tarea de empezar a configurar el sistema a la medida de nuestras necesidades. Este proceso consta, básicamente, de dos partes: configuración de los distintos dispositivos de hardware que tengamos instalados en el ordenador, e instalación del software que vamos a utilizar.

La configuración del hardware del sistema suele ser la parte que cuesta más esfuerzo en general, ya que en cada ordenador encontraremos dispositivos distintos, y por tanto cada ordenador será un mundo. Por lo general, en los sistemas GNU/Linux se puede configurar cualquier dispositivo, por raro que éste sea, aunque en función de su grado de estandarización, esto será más o menos complicado. Pero también es cierto que durante este proceso es cuando más se aprende, ya que, por lo general, configurar un dispositivo implicará siempre ciertas tareas previas, como informarnos exactamente sobre qué tipo de dispositivo es del que disponemos, leer documentación acerca de cómo este tipo de dispositivos se integran en los sistemas GNU/Linux, cómo se hace esta integración para nuestro dispositivo en particular, etc.

Dado que no todos los fabricantes de hardware dan soporte a los sistemas GNU/Linux, y los hay que ni siquiera facilitan la información necesaria para que los desarrolladores de la comunidad puedan escribir el código necesario para poder integrar estos dispositivos al sistema operativo, se recomienda siempre que a la hora de adquirir hardware nuevo, nos informemos sobre cuál es exactamente el producto que deseamos adquirir (información en general mucho más precisa de la que suelen facilitar los proveedores de hardware, y de la cual, a veces, ni disponen), si éste está plenamente soportado, de qué información disponemos para integrar el nuevo producto en nuestro sistema, etc. Los aspectos generales que hay que considerar, a la hora de realizar una nueva adquisición, son a grandes rasgos: el grado de estandarización y calidad del producto. Por lo que a la estandarización se refiere, cuanto más estándar sea el producto, seguro que más usuarios disponen de él y, por tanto, se hace mucho más probable que esté plenamente soportado. En cuanto a calidad del producto, hace ya algunos años que fabricantes de hardware,

ANOTACIONES

para reducir costes, empezaron a sustituir funciones que inicialmente se implementaban vía hardware por soluciones software (siendo el caso más comúnmente conocido de esta práctica, quizás, la de los módems conocidos como winmódems). Esto se traduce, por una parte, en una bajada de rendimiento del sistema, ya que presupone cargar a la CPU con nuevas tareas, para muchas de las cuales ni siquiera ha sido diseñada y, por otra, la necesidad de disponer del software que suplante al hardware eliminado, y que, por lo general, sólo está desarrollado para cierto tipo de sistemas operativos; por esta razón, se recomienda en general rehuir de cualquier producto que se distinga por ser diseñado para un sistema operativo determinado.

A lo largo de cada subsección de configuración se irán comentando algunos aspectos sobre los distintos dispositivos que nos podemos encontrar, y qué problemas llevan implícitos.

Antes de empezar a configurar los distintos dispositivos de nuestro sistema, recordaremos algunas estrategias que nos pueden ser de utilidad para este fin. En primer lugar, mediante el comando `lspci` podemos obtener mucha información acerca de cómo estos dispositivos han sido reconocidos por el sistema durante el proceso de arranque. Si esta información no nos es suficiente, siempre podemos recurrir al directorio virtual `/proc/`, donde queda registrada toda la información acerca del hardware del sistema, entre otras. También pueden ser de utilidad los ficheros de log, ubicados en `/var/log/` (una práctica interesante para ver cómo evoluciona temporalmente el contenido de estos ficheros es utilizar el comando `tail` con el parámetro "`-f`" y redireccionar su salida a una tty que no estemos usando; a modo de ejemplo "`tail -f /var/log/messages > /dev/tty10`").

9.8.1. Configuración del ratón

Al igual que se ha hecho en el taller de KNOPPIX, el deamon que se encargará de gestionar el ratón será gpm. Procedamos pues a instalar el paquete:

```
brau:/etc/apt# apt-get install gpm
```

ANOTACIONES

Al finalizar la instalación, se arranca automáticamente un *script* para asistirnos en la configuración del ratón, los parámetros que debemos pasarle son esencialmente los mismos que le pasamos en su momento en el taller de KNOPPIX, pero si algo fallase, siempre podemos volver a lanzar el programa de configuración mediante el comando `gpmconfig` (por lo general la configuración "`-m /dev/psaux -t ps2`" debería ser válida para la mayoría de *mice* de tres botones PS2). La configuración que utilizará gpm cada vez que arranque se guarda en `/etc/gpm.conf`.

Se recomienda que el ratón tenga tres botones, ya que se acostumbra a asignar funciones a los tres, en especial en los entornos gráficos. Por esta razón, si disponemos de un ratón de tan sólo dos botones, deberemos emular el tercero pulsando los dos a la vez.

Arranque y parada de gpm

Como ya se ha dicho, el programa encargado de gestionar el funcionamiento del ratón es un deamon. Por ello, tanto para lanzarlo como para detenerlo procederemos de la misma manera que se hace con cualquier deamon. Esta subsección servirá de ejemplo para mostrar cómo se arrancan y se paran los deamons.

Tanto el arranque como la parada de un deamon se hace mediante un *script* residente en `/etc/init.d/`. Por lo general, si se invoca este *script* sin ningún parámetro, él mismo nos mostrará una línea de ayuda para orientarnos en su uso. Procedamos pues a parar el deamon gpm:

```
brau:/etc/init.d# ./gpm stop
Stopping mouse interface server: gpm
```

Mediante "`ps aux`" podemos comprobar que, efectivamente, no hay ningún proceso corriendo llamado gpm, y además podemos ver que si movemos el ratón, no se muestra nada por pantalla. Ahora procedamos a arrancarlo:

```
brau:/etc/init.d# ./gpm stop
Starting mouse interface server: gpm
```

Movamos el ratón y observemos cómo en pantalla aparece su puntero. Ahora procedamos a analizar si al arrancar el ordenador este deamon se arrancará automáticamente. El fichero /etc/inittab nos indica en qué *runlevel* se arrancará el sistema operativo: por defecto, el 2; por lo tanto, en dicho archivo deberíamos encontrar una línea como la que sigue:

```
# The default runlevel.
id:2:initdefault:
```

Comprobemos pues si en el directorio /etc/rc2.d/ existe un enlace simbólico a /etc/init.d/gpm:

```
brau:/etc/init.d# ls -l ../rc2.d/ | grep gpm
lrwxrwxrwx    1 root     root          13 feb 21 13:03 S20gpm -> ../init.d/gpm
```

Si este enlace simbólico no existiera, y deseáramos que gpm se arrancase automáticamente durante el proceso de arranque, deberíamos crearlo manualmente mediante "ls -s". Si por el contrario el enlace existiera y no deseáramos que gpm se arrancara, durante el proceso de arranque bastaría con borrar este enlace simbólico; no obstante, no es recomendable borrar los *scripts* de /etc/init.d, ya que son muy útiles de cara a arrancar y detener deamons.

9.8.2. Configuración de módems

Al igual que el resto de hardware, los módems se pueden configurar de modo totalmente manual, pero esta práctica, en general, ha pasado a formar parte del pasado, ya que con el tiempo se han ido desarrollando herramientas suficientemente potentes y fiables que nos pueden ayudar a ahorrarnos la tediosa tarea de configurar un módem manualmente. Una de estas herramientas es pppconfig, que es la que se propone en este texto para configurar nuestro módem.

Pero antes de empezar con la configuración de nuestro módem, hay que poner de manifiesto que no son realmente módems todos los dispositivos que se anuncian o se venden como tales. Como ya se ha dicho, muchos fabricantes, con el objetivo de reducir costes, han ido sustituyendo componentes físicos por software, siendo probablemen-

ANOTACIONES

te los módems los primeros dispositivos que históricamente fueron víctimas de estas prácticas. Hay que prestar especial atención a los módems internos, ya que en realidad son pocos los que incorporan todo el hardware propio de estos dispositivos. Éstos son fácilmente reconocibles por la diferencia de precio con respecto a los falsos módems. Por esta razón, muchos de estos dispositivos se han visto reducidos a meros esclavos de su software (para más información acerca de estos dispositivos y de su integración en GNU/Linux véase:

– http://www.tldp.org/HOWTO/Winmodems-and-Linux-HOWTO.html,
– http://www.tldp.org/HOWTO/Linmodem-HOWTO.html,
– http://www.idir.net/ gromitkc/winmodem.html).

Por esta razón, en general se recomienda utilizar, siempre que sea posible, módems externos. Independientemente de que se disponga de un módem real o no, se recomienda la lectura de http://www.tldp.org/HOWTO/Modem-HOWTO.html.

Dado que vamos a utilizar pppconfig para configurar nuestro módem, si no lo instalamos durante el proceso de instalación del sistema (se puede probar intentando lanzar la aplicación directamente, es decir, tecleando `pppconfig`, o mediante "`dpkg -l | grep pppconfig`"), lo primero que debemos hacer es instalar la aplicación:

```
brau:~# apt-get install ppp pppconfig
```

Una vez nos encontramos en la pantalla principal de la interfaz de instalación, seleccionaremos la opción `Create Create a connection` y en la pantalla siguiente introduciremos el nombre con el que nos referiremos a esta configuración, ya que es posible configurar y gestionar más de una conexión.

Después de asignar un nombre a la nueva configuración, debemos configurar el acceso al DNS, podemos escoger la opción por defecto, asignación estática de DNS, o por asignación dinámica de DNS, si sabemos certeramente que nuestro ISP durante el proceso de conexión nos facilita las direcciones de los DNS. Si escogemos la opción por defecto, se nos pedirá que entremos las IP de los DNS que queremos utilizar, y que se almacenarán en el fichero `/etc/ppp/resolv/`*nombredeconfiguracion*).

ANOTACIONES

En la pantalla siguiente debemos escoger el método de autenticación para establecer la configuración. En general, y salvo casos excepcionales, escogeremos la primera opción, `PAP`. Seguidamente, facilitaremos el nombre de usuario y el *password* de conexión, y seleccionaremos la velocidad de acceso; la que se propone por defecto, `115200`, en la mayoría de conexiones debería funcionar sin ningún problema. Tras especificar si nuestra línea telefónica va por pulsos o por tonos (actualmente, la mayoría ya van por tonos), entraremos el número que se debe marcar y que nos tiene que haber facilitado nuestro ISP.

Llegados a este punto, pppconfig nos propone la ejecución de autodetección del módem. Con el módem en marcha podemos dejar que sea el mismo programa el que detecte a qué puerto está conectado el módem, o lo podemos hacer nosotros manualmente (recordando siempre la correspondencia: primer puerto serie, COM1, `/dev/ttyS0`, segundo puerto serie, COM2, `/dev/ttyS1`, etc).

Una vez hayamos entrado la ttyS a la que está conectado el módem, accederemos a una pantalla de resumen de los datos que hemos entrado, la cual también nos ofrece la posibilidad de establecer opciones avanzadas (por lo general, no necesarias). Si los datos son correctos, podemos escoger la opción `Finished Write files and return to main menu`, y tras confirmar la operación, de retorno al menú principal de la interfaz, si no queremos configurar ninguna otra conexión, escogeremos la opción `Quit Exit this utility` para volver a la línea de comandos. Una vez en la línea de comandos, podemos comprobar que los datos se han guardado correctamente en `/etc/ppp/peers/`*nombredeconfiguración*.

Para los usuarios de conexiones PPP también puede ser interesante instalar el paquete `pppstatus` para monitorizar el tráfico de la conexión, entre otros.

Establecimiento y finalización de conexión: `pon, poff`

Para establecer la conexión, bastará con teclear la instrucción `pon` seguido del nombre de conexión que queramos utilizar; si sólo hemos configurado una conexión, no será necesario especificar su

ANOTACIONES

nombre. En principio, si no se restringe el uso, `pon` puede ser ejecutado por cualquier usuario.

Para finalizar la conexión bastará con ejecutar el comando `poff`.

9.8.3. Configuración de módems DSL

De la misma manera que se ha hecho en la sección anterior para la configuración de módems tradicionales, utilizaremos una herramienta para configurar los módems DSL: pp-poeconf. Pero antes de empezar se recomienda la lectura de http://www.tldp.org/HOWTO/DSL-HOWTO/ y de http://www.tldp.org/HOWTO/ADSL-Bandwidth-Management-HOWTO/.

9.8.4. Configuración de tarjetas de red

Si por el motivo que sea no hemos configurado la tarjeta de red durante el proceso de instalación, ahora es el momento de hacerlo. Aunque prácticamente todas las tarjetas de red están soportadas en los sistemas GNU/Linux, ya que éstas son una pieza clave para un sistema operativo orientado a redes, una vez más se recomienda el uso de hardware lo más estándar posible para no tener complicaciones a la hora de configurarla. Puede que nuestra tarjeta venga soportada de dos formas distintas: la primera es que su *driver* haya sido directamente compilado dentro del propio *kernel*, y la segunda, es que el *driver* haya sido compilado en forma modular, para que se cargue posteriormente.

La forma más sencilla de saber si el *driver* de nuestra tarjeta de red ha sido compilado dentro del propio *kernel* es analizando el mensaje de retorno de `dmesg` tal y como se ha hecho en 5.4.3. La configuración completa del *flavor* **bf24** se puede encontrar en ftp://ftp.debian.org/debian/dists/woody/main/disks-i386/current bf2.4/kernel-config entre otros.

Si después de analizarse detalladamente la salida del comando `dmesg` llegamos a la conclusión de que el *driver* para nuestra tarjeta no ha sido cargado, podemos ejecutar el comando `modconf`, que sirve para cargar módulos al *kernel* que han sido compilados junto

ANOTACIONES

a él, y comprobar si el *driver* para ésta aparece en la subsección `kernel/drivers/net`. Si es así, bastará con seleccionarlo para cargarlo en el *kernel*.

Si nuestra tarjeta de red no está soportada por defecto, deberemos recurrir a la recompilación del *kernel*.

Una vez hecho esto, editaremos el fichero `/etc/network/interfaces` para pasar los parámetros correspondientes a nuestra red a la tarjeta. Una posible configuración sería:

```
# /etc/network/interfaces -configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

# The first network card this entry was created during the Debian installation
# (network, broadcast and gateway are optional)
auto eth0
iface eth0 inet static
      address 158.109.69.132
      netmask 255.255.0.0
      network 158.109.0.0
      broadcast 158.109.255.255
      gateway 158.109.0.3
```

Se recomienda la lectura de http://www.tldp.org/HOWTO/Networking-Overview-HOWTO.html, también http://www.fokus.gmd.de/linux/HOWTO/Net-HOWTO/ y el man de `interfaces`. Si no disponemos de tarjeta de red y quisiéramos hacer pruebas, siempre podemos recurrir al módulo `dummy`; en este caso, el dispositivo en vez de llamarse `eth0` se llamaría `dummy0`.

Si se quiere configurar más de una tarjeta de red en el mismo ordenador (práctica muy habitual en *gateways*, entre otros), es necesario pasar los parámetros correspondientes al *kernel* durante el proceso de arranque (utilizando `append` en Lilo, por ejemplo) para evitar conflictos entre dispositivos.

ANOTACIONES

Arranque y parada de servicios de red: `ifup`, `ifgdown`

La reinicialización de todos los servicios de red (los de `/etc/network/interfaces`) se puede hacer mediante el *script* `/etc/init.d/networking` con el parámetro `restart`.

`ifup` se utiliza para arrancar los servicios de red de una interfaz determinada, e `ifdown` para pararlos. Así pues, para la configuración anterior, si quisiéramos detener y volver a arrancar los servicios de `eth0`, lo que haríamos es lo siguiente (se utiliza el comando `ifconfig` para comprobar los resultados):

```
brau:~# ifconfig
eth0 Link encap:Ethernet HWaddr 00:01:02:B4:3A:61
     inet addr:158.109.69.132
     Bcast:158.109.255.255 Mask:255.255.0.0
     UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
     RX packets:36409683 errors:0 dropped:0 overruns:221 frame:0
     TX packets:35938 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:100
     RX bytes:1489273710 (1.3 GiB) TX bytes:20116974 (19.1 MiB)
     Interrupt:5 Base address:0x9400

lo   Link encap:Local Loopback
     inet addr:127.0.0.1 Mask:255.0.0.0
     UP LOOPBACK RUNNING MTU:16436 Metric:1
     RX packets:823 errors:0 dropped:0 overruns:0 frame:0
     TX packets:823 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:0
     RX bytes:3169619 (3.0 MiB) TX bytes:3169619 (3.0 MiB)

brau:~# ifdown eth0
brau:~# ifconfig
lo   Link encap:Local Loopback
     inet addr:127.0.0.1 Mask:255.0.0.0
     UP LOOPBACK RUNNING MTU:16436 Metric:1
     RX packets:823 errors:0 dropped:0 overruns:0 frame:0
     TX packets:823 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:0
     RX bytes:3169619 (3.0 MiB) TX bytes:3169619 (3.0 MiB)
```

```
brau:˜# ifup eth0
brau:˜# ifconfig
eth0 Link encap:Ethernet HWaddr 00:01:02:B4:3A:61
     inet addr:158.109.69.132 Bcast:158.109.255.255 Mask:255.255.0.0
     UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
     RX packets:36420981 errors:0 dropped:0 overruns:221 frame:0
     TX packets:35965 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:100
     RX bytes:1490867554 (1.3 GiB) TX bytes:20118868 (19.1 MiB)
     Interrupt:5 Base address:0x9400

lo   Link encap:Local Loopback
     inet addr:127.0.0.1 Mask:255.0.0.0
     UP LOOPBACK RUNNING MTU:16436 Metric:1
     RX packets:823 errors:0 dropped:0 overruns:0 frame:0
     TX packets:823 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:0
     RX bytes:3169619 (3.0 MiB) TX bytes:3169619 (3.0 MiB)
```

9.8.5. Configuración de impresoras

Tener configurada la impresora puede ser de gran utilidad, ya que esto nos permitirá, entre otras cosas, imprimir los ficheros de `man`, los de configuración, etc. para poderlos estudiar más detenidamente sobre formato papel.

Por lo que se refiere al tipo de impresora preferible, en un entorno doméstico sería una que usara el puerto paralelo (`/dev/lpX`, normalmente `/dev/lp0`) con el objetivo de garantizar que se trata de una impresora no dependiente de su software, ya que la mayoría de las nuevas que están apareciendo en el mercado, especialmente las de puerto USB, son impresoras generalmente diseñadas para un sistema operativo determinado. Estas impresoras son popularmente conocidas como *winprinters* (y su génesis es parecida a la de los denominados *winmodems*). En un entorno profesional, lo óptimo sería contar con una impresora que incorporara su propia interfaz de red y que, por lo tanto, fuera un nodo más de ésta. Para más información acerca de impresoras y su integración en los sistemas operativos GNU/Linux, se recomienda visitar la página `http://www.linuxprinting.org`, donde encontraremos un

ANOTACIONES

listado exhaustivo de las impresoras existentes en el mercado, y su grado de soporte.

La topología general del sistema de impresión bajo GNU/Linux es la de cliente-servidor. El servidor, `lpd` (en el sistema BSD), es tipo daemon, y en consecuencia lo manipularemos como tal. El archivo de configuración del servidor es `/etc/printcap`. En él podemos configurar tantas impresoras como deseemos. Ante la presencia de más de una impresora configurada, los clientes usan, normalmente, el parámetro "-p" seguido del nombre de la impresora, para referirse a ésta.

Como hemos visto, configurar correctamente una impresora puede ser un tanto difícil. Por esta razón, han aparecido diversos proyectos para desarrollar interfaces que harán más amena su configuración, algunos de los cuales aportan su propio servidor de impresión y sus propias herramientas cliente.

Impresión de ficheros de texto

Los formateadores son programas que se utilizan, principalmente, para transcribir ficheros en formato texto a formato PostScript (el lenguaje PostScrip históricamente ha tenido más implementación en el campo de la impresión). Estos programas nos pueden ser de utilidad, ya que nos permiten pasar a formato papel la ayuda de los comandos, ficheros de configuración, etc. para poderlos estudiar de una forma más cómoda. Entre éstos encontramos `mpager` (dentro del paquete con el mismo nombre), o `enscrip` (también empaquetado con este mismo nombre, y más potente que el anterior). A continuación, se detallan un par de líneas para ejemplificar su uso:

```
man man | mpage -2 -o |lpr
```

Mediante esta línea redireccionamos la salida de man de `man` a `mpage`, que le da formato a dos columnas por hoja, sin márgenes, y redireccionamos su salida al cliente de impresión `lpr`:

```
endscript /etc/fstab -B -fTimes-Roman7 -r
```

Con esta línea haremos que se imprima el contenido del fichero `/etc/fstab` sin cabecera, utilizando el tipo de carácter Times-Roman de tamaño 7 y de forma apaisada.

ANOTACIONES

9.8.6. Configuración de tarjetas de sonido

Debido a la gran cantidad de tarjetas de sonido existentes en el mercado, se hace casi imposible dar una descripción de cómo configurarlas todas. Se recomienda la lectura de http://www.tldp.org/HOWTO/Sound-HOWTO/ y la visita a las páginas de los dos proyectos más destacados en cuanto a sonido bajo GNU/Linux se refiere: http://www.opensound.com/ y http://www.alsa-project.org/.

A continuación se expondrá el modo de proceder para configurar una tarjeta de sonido bastante común: SoundBlasterPCI (chipset ES1371). Para este tipo de tarjeta, el comando `lspci` nos devolverá una línea como la siguiente:

```
00:0d.0 Multimedia audio controller: Ensoniq 5880
AudioPCI (rev 02)
```

En primer lugar, cargaremos el módulo correspondiente a esta tarjeta de sonido mediante el comando `modconf`, `kernel/drivers/sound`, `es1371`.

Seguidamente, crearemos el grupo `audio` en `/etc/group` e incluiremos en él a todos los usuarios que deseamos que tengan acceso al dispositivo de sonido (si deseamos que todos los usuarios tengan acceso a él, podemos obviar este paso, y dar todos los permisos a los ficheros `/dev/dsp` y `/dev/mixer`); a continuación asociaremos los ficheros `/dev/dsp` y `/dev/mixer` al nuevo grupo creado.

Con esto ya tenemos configurada la tarjeta de sonido. Ahora podemos comprobarlo direccionando un fichero de audio directamente a `/dev/dsp`, como sugiere http://www.tldp.org/HOWTO/Sound-HOWTO/ o esperar a tener el entorno gráfico configurado para poder instalar aplicaciones de audio que corren sobre él.

9.9. Conclusión

En este taller hemos aprendido a trabajar con el sistema de paquetes de Debian, hecho fundamental, ya que esto nos ha permitido aprender a instalar, desinstalar y a gestionar aplicaciones. También hemos

ANOTACIONES

aprendido a configurar distintos dispositivos de hardware, y con ello una cosa básica: que con GNU/Linux esto no es tan sencillo como en otros sistemas operativos, ya que requiere, por lo general, tener un conocimiento más profundo tanto del propio dispositivo como del sistema en sí. Pero en compensación podemos asegurar que, una vez configurado un dispositivo, éste funcionará perfectamente y no deberemos preocuparnos más de él. Y todo ello teniendo en mente que aún no hemos configurado el entorno gráfico (probablemente una de las misiones más complicadas en un entorno GNU/Linux). No obstante, y para animar a los que a estas alturas puedan pensar que adentrarse en el mundo GNU/Linux ha sido una mala idea, una misión fallida y una pérdida de tiempo, nos gustaría citar un párrafo del libro Debian GNU/Linux 2.1 de Mario Camou, John Goerzen y Aaron Van CouWenberghe, parte I, capítulo 1, *Why Linux Is Better*:

"Windows NT, however, learned to speak the language of the Internet just a few years ago. It is not refined by any stretch imagination, but plagued with bugs and inconsistences. Bussinesses need a solution that they can put online and expect to remain available 100% of the time, but Windows NT cannot meet this need. On the contrary, a Windows NT administrator's most common job is crash recovery; he wastes too much time doing busywork, such as booting servers that have gone down[...]"

Aaron Van CouWenberghe

ANOTACIONES

10. Arquitectura X-Window

10.1. ¿Qué es X-Window?

X-window es una arquitectura de ventanas diseñada a mediados de los ochenta para poder disponer de un entorno gráfico en estaciones de trabajo. A diferencia de otros entornos de ventanas, la arquitectura X-Window se diseñó para ser independiente de plataforma, de manera que se pudiera instalar en cualquier ordenador que corriera un sistema tipo UNIX. Aunque la arquitectura de ventanas X-Window ha tenido una dilatada historia en la que se han utilizado diferentes tipos de licencias, varias implementaciones y muchos equipos de desarrollo diferentes, actualmente se utiliza, mayoritariamente, la implementación que ha desarrollado el proyecto XFree86 Inc, llamada **XFree86**. Esta implementación se distribuye con licencia *open source*, que aunque no es exactamente igual que la GPL o sus variantes, tiene características parecidas que permiten acceder a su código fuente, su redistribución, etc. Por este motivo, en la mayoría de distribuciones de GNU/Linux (y cada vez más en otros sistemas operativos -como Mac OS X-) se incorpora esta implementación de X-Window.

Contenido complementario

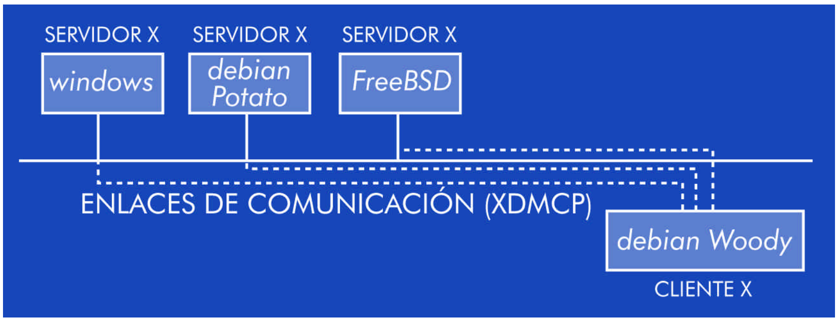
Una arquitectura de ventanas (o sistema de ventanas) es un entorno gráfico que nos proporciona la posibilidad de tener diferentes aplicaciones localizadas en diferentes regiones de la pantalla, generalmente delimitadas por algún tipo de ventana. Estos entornos suelen proporcionar mecanismos para el desplazamiento y manipulación de estas ventanas de forma que el trabajo pueda ser más interactivo y ameno.

X-Window está diseñado con una arquitectura cliente/servidor. Este tipo de arquitectura significa que el software está estructurado en dos partes totalmente independientes (cliente y servidor) que se comunican a partir de un enlace de comunicación. Aunque esto implica que el diseño y la codificación es un poco más compleja, esta arquitectura proporciona una flexibilidad total en el sentido que cliente y servidor pueden estar ubicados en diferentes sitios y utilizando diferentes plataformas y/o sistemas operativos. Además, podemos aprovechar muchísimo más un mismo cliente, ya que éste podrá dar servicio a más de un servidor a la vez. De esta forma, los ordenadores servidores pueden trabajar con un entorno gráfico y los recursos del cliente. Naturalmente, esta arquitectura también nos permite trabajar con X-Window de forma local en la máquina donde está situado el cliente, aunque no es indispensable.

Los componentes de los que está compuesto X-Window son: cliente, servidor y enlace de comunicación. Cliente y servidor están diseñados para ser independientes de plataforma y, en el caso del enlace de comunicación, para ser independiente del protocolo de red.

De este modo, podemos utilizar X-Window en cualquier tipo de escenario; por ejemplo, podríamos tener el servidor instalado en un ordenador con Windows^{TM}, conectándose a un cliente con GNU/Linux y utilizar como canal de comunicación Internet (protocolo IPv4). Aunque la configuración de cada uno de estos componentes (sobre todo el cliente) sí que depende, en cierto modo, de la plataforma donde está instalado, el enlace de comunicación nos permite aislar los componentes, dándoles un lenguaje propio para su entendimiento.

Este enlace utiliza un protocolo propio denominado XDMCP (X *Display Manager Control Protocol),* que está en un nivel superior al de la red de comunicación utilizada (por eso es independiente de red).



En esta arquitectura, el servidor está ideado para recoger los eventos que se producen por los dispositivos de entrada como el teclado, el ratón, etc. y enviarlos al cliente. El cliente procesa estos eventos y responde al cliente, que muestra los resultados en los dispositivos de salida (generalmente el monitor). Aunque la primera impresión que puede sugerirnos este diseño es que el tiempo de respuesta debe ser muy lento, el protocolo XDMCP está especialmente diseñado para proporcionar un enlace rápido entre cliente y servidor, de forma que se pueda trabajar realmente de forma interactiva. En los únicos escenarios en que podemos notar este inconveniente es en conexiones remotas utilizando redes de comunicaciones lentas.

ANOTACIONES

En resumen, pues, las principales características y funciones de cada uno de los componentes de X-Window son las siguientes:

Cliente	Gestión de diferentes servidores simultáneamente
	Dependiente de plataforma
	Procesamiento de las aplicaciones
Servidor	Control del display del usuario
	Independiente de plataforma
	Procesamiento de los dispositivos de entrada
Enlace	Diseñado para poder trabajar interactivamente
	Pensado para minimizar el tráfico en la red
	Transparente (independiente de red)

A medida que las tarjetas gráficas han ido evolucionando, cada vez más aplicaciones y juegos necesitan de un procesamiento en 2D o 3D más rápido. Si bien la arquitectura de ventanas X-Window aporta muchas ventajas, cuando queremos utilizar este tipo de aplicaciones el diseño cliente/servidor no es el más adecuado, ya que no aprovechamos las funciones de procesamiento 2D y 3D extremadamente rápido de las tarjetas gráficas instaladas en el servidor. Para solucionar este problema, a partir de 1998 apareció una tecnología llamada DRI (*Direct Rendering Infrastructure*), que permite aprovechar los chips de procesamiento de las tarjetas para ahorrar trabajo al cliente X-Window. De esta forma, continuamos teniendo todas las ventajas de X-Window aprovechando los elementos específicos de las tarjetas gráficas.

A diferencia de otros sistemas operativos donde el entorno gráfico está íntimamente integrado con el resto de funciones, la arquitectura X-Window es totalmente independiente del operativo y no nos limita a ningún GUI (*Grafic User Interface*) determinado. De hecho, la arquitectura sólo nos proporciona herramientas gráficas de bajo nivel para manipular la salida del monitor. Estas herramientas están incluidas en la librería **Xlib** y principalmente son funciones para crear y manipular ventanas, operaciones con fuentes de caracteres, detección de eventos de usuario y operaciones gráficas. Con estas funciones podemos dotar a nuestras aplicaciones del *look and feel* que queramos, crear nuevos GUI, . . . De hecho, esto supuso un trabajo adicional para los primeros desarrolladores de aplicaciones en Xwindow, ya que además de programar la aplicación tenían que desarrollar sus propias librerías para la creación de menús, iconos, etc. A medida que X-Window fue creciendo, fueron apareciendo lo que llamamos **toolkits**, que son librerías generalmente implementadas con Xlib y que proporcionan un GUI particular. De esta

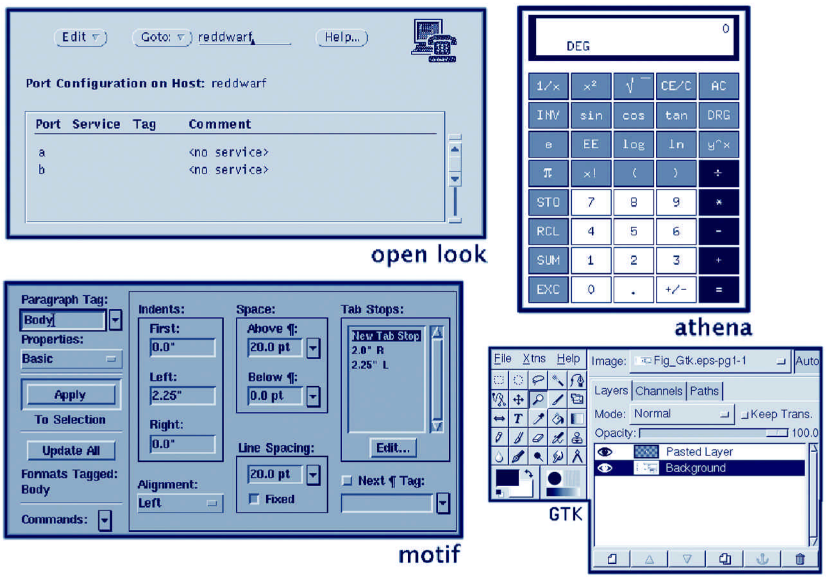
Contenido complementario

El *look and feel* es el diseño utilizado para los botones, barras de desplazamiento, menús, etc. de un entorno gráfico o una aplicación.

ANOTACIONES

manera, al diseñar una aplicación podemos utilizar alguno de estos *toolkits* que ya proporcionan las herramientas estándar para crear menús, botones, gestionar los *cut and paste*, . . . y centrarnos en programar la aplicación en sí. El no marcar ningún *look and feel* ha sido otra de las claves del éxito de la arquitectura X-Window, ya que cada fabricante o desarrollador de software ha podido diseñarse uno propio, marcando la diferencia con los demás.

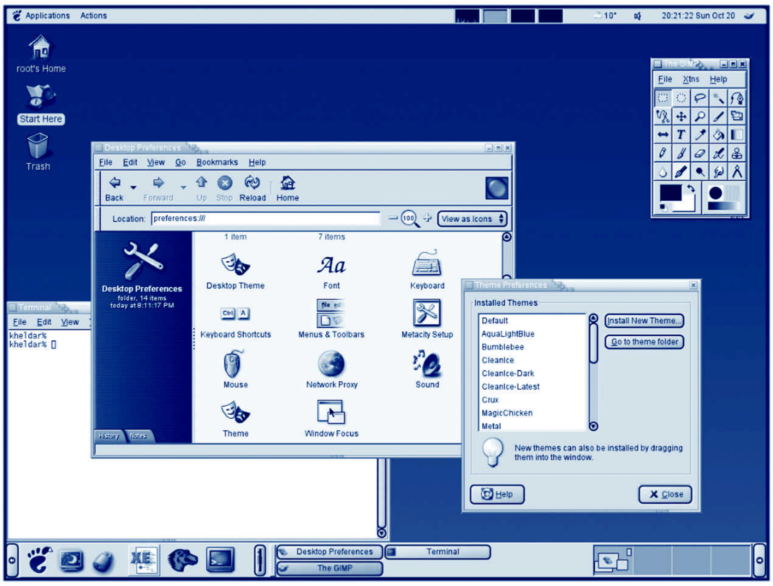
Aunque existen muchos *toolkits* diferentes, en la siguiente figura podemos ver algunos de los más populares que se han utilizado a lo largo de la historia de X-Window:



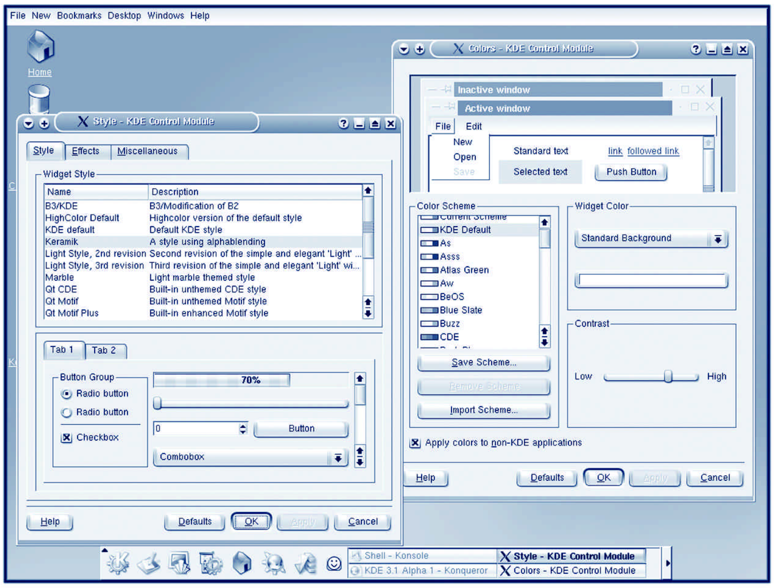
El *window manager* es un servidor especial de X-Window que se encarga de gestionar todas las ventanas, los escritorios, las pantallas virtuales, etc. Naturalmente, todas las aplicaciones pueden funcionar con cualquier *window manager*, ya que éste sólo se encarga de gestionar la ventana donde está ubicado el programa. Aunque la programación de un *window manager* es muy diferente que la de una aplicación, también se suelen utilizar *toolkits* particulares que proporcionan un *look and feel* determinado. Actualmente existen decenas de *window managers* diferentes (wmaker, sawmill, olvwm, . . .), siendo el mismo usuario quien puede elegir el que más le guste.

Otro tipo de software muy relacionado con X-Window es el que se encarga de proporcionar un **entorno integrado** para las aplicaciones, el escritorio, las herramientas de administración del sistema, etc. Los más

populares que existen actualmente son el KDE (the K *Desktop Environment*) y el GNOME (GNU *Object Model Environment*). Los dos proporcionan un *toolkit* particular, un entorno de escritorio con muchísimas funcionalidades y configuraciones diferentes y una lista de aplicaciones integradas que cada vez va creciendo más. La mayoría de distribuciones de GNU/Linux y UNIX proporcionan alguno de estos dos entornos de escritorio por ser muy amigables y proporcionar herramientas y software propio de gran calidad que ayudan en gran medida al usuario para configurar el sistema y el mismo escritorio. Los dos pueden funcionar con cualquier *window manager* que cumpla con una serie de características básicas. En la siguiente figura podemos ver el aspecto de los dos:



GNOME



KDE

Finalmente, otro tipo de aplicación que se utiliza en X-Window es el ***session manager***, que son una serie de programas que permiten guardar la configuración de una determinada sesión de usuario para que al arrancar de nuevo X-Window se carguen las aplicaciones que tenga configuradas. Generalmente, en los entornos integrados ya se incorporan estas herramientas de forma automática; si no, podemos recurrir al que la misma infraestructura de X-Window proporciona: el `xsm`.

Actividades

14. Leer la historia de X-Window en el artículo: http://www.linux-mag.com/2001-12/xfree86 01.html

15. Ver algunos de los *window manager* y entornos de escritorio existentes en: http://www.xwinman.org

10.2. Configuración

Actualmente, las versiones de la implementación XFree86 que más se utilizan son las 4.X, cuya configuración veremos en esta sección. Si bien la mayoría de tarjetas gráficas del mercado ya están soportadas, es posible que desde el momento de aparición en el mercado de una nueva tarjeta hasta que se da soporte en X-Window pasen unas semanas o unos pocos meses. De todos modos, cada vez más los mismos fabricantes están dando soporte a GNU/Linux y, en algunos casos ya están proporcionando sus propios *drivers* para este sistema operativo. Aun así, antes de comprar una nueva tarjeta gráfica, siempre es recomendable comprobar si está disponible algún tipo de *driver* para la distribución que estemos utilizando.

Para instalar XFree86 en nuestro ordenador lo primero que deberemos hacer es bajarnos los paquetes que contienen las herramientas básicas y el software para el cliente y el servidor. Generalmente, estos paquetes se suelen denominar `xfree86-common`, `xfree86server`, etc. y llevan implícitos varias dependencias de fuentes y algunas utilidades básicas para el manejo de X-Window. Una vez instalados estos paquetes, debemos configurar adecuadamente los dispositivos de los

ANOTACIONES

Contenido complementario

Algunos programas típicos de configuración de X-Window son el `xf86config` o el `XF86Setup`.

que disponemos para poder arrancar correctamente el cliente y servidor X-Window. Según la distribución que utilicemos, se hace uso de uno u otro programa o, en algunos casos, con la misma instalación de los paquetes ya se lanza una pequeña aplicación de configuración. No obstante, esta configuración siempre debe contener unos determinados pasos, que detallamos a continuación clasificados según el dispositivo que hay que configurar:

1) Tarjeta gráfica

- *Driver*: las diferentes familias de tarjetas gráficas llevan unos microprocesadores específicos y utilizan unas funciones determinadas para realizar sus operaciones. Por esta razón, debemos indicar el *driver* adecuado para nuestra tarjeta. Si no lo sabemos, podemos instalar algún tipo de aplicación para la detección de hardware automático; si utilizamos, por ejemplo, el `discover`, podemos saber qué *driver* necesita nuestra tarjeta con el comando "`discover --xdriver video`".

- Identificador: el identificador de la tarjeta puede ser cualquier nombre con el que queremos referirnos a nuestra tarjeta. Este identificador es utilizado internamente para poder referenciar adecuadamente las tarjetas que tenemos instaladas en el sistema.

- Cantidad de memoria: según la cantidad de memoria de la tarjeta, podremos inicializar los gráficos con más o menos resolución y con profundidades de color más o menos elevadas. Aunque no es imprescindible indicar esta cantidad (el sistema lo detecta automáticamente) sí que es recomendable especificarla en la configuración.

- Utilización del *framebuffer* del núcleo: el *frambuffer* del núcleo es un *driver* especial de Linux que permite realizar algunas operaciones sobre X-Window. Aunque su utilización no es obligatoria, generalmente se utiliza para que el servidor de X-Window se pueda comunicar directamente con el núcleo del sistema. De todos modos, si nos diera algún problema, siempre podemos desactivarla.

ANOTACIONES

2) Teclado

- Regla XKB: para que el servidor de X-Window pueda manejar correctamente el teclado, necesita saber qué reglas aplicar sobre él. Para la mayoría de teclados estándar de los PC, se utiliza la regla "`xfree86`" y para las estaciones de trabajo Sun, se suele utilizar la regla "`sun`".

- Modelo de teclado: el modelo de teclado generalmente se suele identificar a partir del número de teclas que tiene. Los teclados de los PC estándar que tienen las teclas de menú y logo suelen tener 104 teclas (los identificamos con el nombre "`pc104`"). Los teclados que no llevan estas teclas se identifican como de 101 teclas ("`pc101`").

- *Keyboard layout*: en esta sección debemos identificar el país del teclado con su referencia ISO 3166. En el caso de España es "`es`", para Francia "`fr`", etc.

- *Keyboard options*: opción para personalizar algunas de las teclas del teclado.

3) Ratón

- Puerto: el puerto del ratón es la conexión que utiliza para comunicarse con el ordenador. Cuando compramos el ratón, siempre se indica si es de tipo PS/2, serie, etc. En el caso de que sea de tipo PS/2, el puerto será `/dev/psaux`, para los ratones serie el puerto será `/dev/ttyS0` (COM1), `/dev/ttyS1` (COM2) y consecutivamente.

- Tipo: para especificar el tipo del ratón, se suele proporcionar una lista de la que debemos escoger el que más se ajuste a nuestro modelo y fabricante. Generalmente, sabiendo el modelo del ratón ya podremos escoger adecuadamente la opción que le corresponde.

- Emulación de 3 botones: en el caso de que nuestro ratón sólo tenga 2 botones, se proporciona la posibilidad de emular el tercero

ANOTACIONES

(el del medio) apretando los dos simultáneamente. Si nuestro ratón no tiene el botón del centro, es recomendable activar esta opción porque algunos programas de X-Window necesitan que el ratón tenga los 3 botones.

4) Monitor

- Identificador: igual que en el caso de la tarjeta gráfica, la identificación del monitor sirve para que el sistema lo pueda referenciar internamente. Le podemos poner el nombre que queramos.

- Monitor tipo LCD: en la mayoría de procesos de configuración se nos preguntará si nuestro monitor es de tipo LCD (pantalla TFT). Es importante responder correctamente a esta pregunta porque el manejo de un tipo u otro de monitor varía considerablemente.

- Características: en la configuración de características se preguntará las resoluciones máximas que puede obtener nuestro monitor, la frecuencia de refresco, etc. Aunque según el programa utilizado para configurar X-Window se plantearán más o menos preguntas de este estilo, es importante tener a mano la información del monitor y contestar adecuadamente para poder aprovechar al máximo las características del mismo.

- Resoluciones disponibles: en este paso debemos señalar qué resoluciones queremos poder mostrar en nuestro monitor cuando iniciemos X-Window. También es habitual que se nos pregunte la profundidad de color que queremos utilizar por defecto; lo más recomendable es utilizar una alta (16 o 24 bits) para poder ver nítidamente todos los colores.

Una vez contestadas estas preguntas, que pueden ser más o menos según el programa que utilicemos, toda la configuración se guarda en el fichero `/etc/X11/XF86Config-4`.

Este fichero está organizado en las diferentes secciones que hemos ido viendo y, recurriendo a su manual, veremos que tenemos muchísimas más posibilidades que nos dan una flexibilidad total para configurar de la forma como queramos nuestras X-Window. Para probar si realmente funcionan, podemos ejecutar "`X`", con lo cual debería

ANOTACIONES

Contenido complementario

Cuando utilizamos `startx` o `xinit` se ejecutan las instrucciones del fichero `/etc/X11/Xsession`. Si en el directorio home del usuario que iniciara X-Window hubiera un fichero `.xsession`, se ejecutarían las instrucciones de éste en lugar del otro.

aparecernos una pantalla con cuadros blancos y negros muy pequeños y el puntero del ratón como una X (para salir de la misma podemos utilizar CTRL+ALT+BACKSPACE).

Si tenemos instalado algún *window manager,* lo más habitual para arrancar X-Window es utilizar alguno de los *shell scripts* `xinit` o `startx`. Éstos se encargan de lanzar el *window manager* configurado y realizan algunas otras acciones necesarias para inicializar correctamente X-Window. Una vez tenemos la pantalla en modo gráfico, podemos cambiar la resolución de la misma con las teclas CTRL+ALT++ y CTRL+ALT+-, o volver a las consolas de texto con CTRL+ALT+F1, CTRL+ALT+F2, etc. (con CTRL+ALT+F7 volveríamos a la gráfica).

Otra característica importante en la configuración de X-Window es la de la configuración de los **Xwrappers**. Los Xwrappers nos permiten controlar qué usuarios pueden iniciar una sesión con X-Window. En el fichero `/etc/X11/Xwrapper.config` se encuentra la directiva "`allowed users`", con la cual especificamos quién está autorizado para arrancar X-Window con los valores:

- "`console`": cualquier usuario que esté en un consola local puede iniciar X-Window.
- "`rootonly`": sólo el *root* puede iniciar X-Window.
- "`anybody`": cualquier usuario del sistema puede iniciar X-Window (aunque no esté conectado localmente).

Esto es muy útil, sobre todo, al administrar un servidor en el que generalmente no se permite que los usuarios trabajen con el entorno gráfico por el gasto de recursos que ello supone.

10.3. X *display manager*

En la sección anterior hemos visto cómo configurar X-Window de forma local. Tal como hemos ido comentando a lo largo del capítulo, la arquitectura de ventanas X-Window nos permite que cliente y ser-

ANOTACIONES

vidor estén instalados en diferentes ordenadores. Para configurar nuestro ordenador de modo que realice las funciones de un cliente X-Window, debemos instalar algún tipo de X *display manager*. Estos programas abren un puerto de comunicaciones con el cual los clientes se pueden comunicar con el cliente y trabajar con X-Window de forma remota. Aunque existen muchos programas de este tipo, uno de los primeros que apareció, y en el que se basan muchos otros, es el `xdm`.

Los X *display manager* pueden actuar tanto de forma local como remota. Entre otras funciones, lo que hacen es mostrar una pantalla (en el entorno gráfico) para que el usuario se identifique con su *login* y contraseña. Funcionan como cualquier otro daemon del sistema, de forma que su inicio y parada se puede configurar como queramos (utilizando los niveles de ejecución que el sistema proporciona). Debemos tener en cuenta que si lo configuramos para que funcione de forma local, al arrancar el sistema nos encontraremos con la pantalla de identificación gráfica y no las consolas a las que estábamos acostumbrados (aunque continúan estando disponibles). Con esta configuración ya no podremos utilizar `startx` o `xinit` para inicializar X-Window, ya que por defecto estarán ejecutándose.

Cuando instalemos `xdm`, todos sus ficheros de configuración se dejarán en el directorio `/etc/X11/xdm`. Vamos a repasar qué contiene cada uno de estos ficheros:

`xdm-config`	Localización de los archivos de configuración de xdm.
`xdm.options`	Opciones globales de configuración.
`Xaccess`	Definición de los equipos remotos a los que dejamos acceder.
`Xservers`	Servidores locales de xdm.
`Xresources`	Configuración de la pantalla de login: colores, fuentes, tamaño, etc.
`Xsetup`	*Script* que se ejecutará cuando se arranque xdm.
`Xstartup`	*Script* que se ejecutará cuando un usuario entre en XWindow. Se suelen poner acciones relacionadas con el xdm.
`Xsession`	*Script* que se ejecutará al entrar en una sesión de usuario. Se suelen poner acciones especiales para los usuarios, aunque también se suele llamar a la ejecución del fichero `/etc/X11/Xsession`.
`Xreset`	*Script* que se ejecutará al acabar una sesión de usuario.

ANOTACIONES

La configuración de los servidores locales la encontramos en el fichero Xservers. Si quisiéramos desactivar el servidor local, podríamos comentar todas las líneas de este archivo. De este modo, aunque tuviéramos instalado un cliente de X-Window, por defecto no se iniciaría en la máquina local. Si por el contrario quisiéramos instalar más de uno, podríamos editar el fichero y añadir directivas como las que siguen:

```
:0 local /usr/X11R6/bin/X :0 vt7
:1 local /usr/X11R6/bin/X :1 vt8
```

Estas dos directivas indican que queremos 2 instancias de X-Window, una en la consola 7 (“vt7”) y la otra en la 8 (“vt8”), accesibles con CTRL+ALT+F7 y CTRL+ALT+F8 respectivamente. Fijémonos cómo cada directiva incluye un “:0” o “:1”, que hacen referencia a la instancia de X-Window que manejan. Por defecto, siempre se utiliza la 0, pero al querer más de un servidor local debemos referenciarlo de esta forma. Al final de cada una de estas líneas podríamos añadir parámetros especiales para cada servidor de X-Window (en “man X” encontramos todos los posibles), como la profundidad de color que queremos para cada uno, la resolución de la pantalla, etc. De esta manera, podríamos trabajar con diferentes sesiones de X-Window abiertas tal como hacíamos con las consolas.

Generalmente, la configuración por defecto de xdm no permite conexiones remotas por razones de seguridad. Si quisiéramos activar estas conexiones, podríamos editar el fichero Xaccess y, utilizando la sintaxis que se nos indica, añadir los servidores a los que permitimos dar este servicio. También deberíamos comentar la línea “DisplayManager. requestPort: 0” del fichero xdm-config, que por defecto inutiliza todas las conexiones que se reciben. Una vez realizados estos cambios, reiniciando el daemon de xdm, el cliente ya estaría preparado para servir X-Window a cualquier servidor que se lo pidiera.

Para las máquinas donde sólo queremos instalar el servidor de X-Window, deberíamos instalar X-Window tal como veíamos en el apartado anterior y utilizar el comando “X -query IP”, donde la IP debería ser la del cliente. Del mismo modo que cuando teníamos más de un servidor X-Window en una máquina local, si en la máqui-

Contenido complementario

Si en el fichero de Xaccess hay una línea con el carácter “*”, indica que dejamos que cualquier servidor se conecte al X-Window del servidor. Utilizar X-Window de forma remota sin ningún tipo de encriptación puede suponer un fallo de seguridad, con lo que es muy recomendable informarse adecuadamente antes de utilizarlo.

Contenido complementario

Xnest es un servidor de X-Window que nos permite abrir en una ventana otra instancia de X-Window.

ANOTACIONES

na ya tuviéramos otra instancia de X-Window ejecutándose, deberíamos utilizar "`X -query IP :1`" para la segunda instancia, "`:2`", para la tercera y consecutivamente.

ANOTACIONES

11. Taller de X-windows

11.1. Introducción

En el segundo taller se sentaron las bases para la correcta manipulación y gestión de paquetes, y aprendimos a configurar algunos dispositivos. No obstante, y debido a su complejidad, no se abordó el tema de la configuración de la tarjeta gráfica, o mejor dicho, de la instalación del entorno gráfico X. Debido a la complejidad de su estructura y configuración, se ha optado por dedicar un taller monográfico sobre el X Window System. En este taller aprenderemos a instalar, configurar y personalizar este sistema. Pero no se pretende hacer un repaso exhaustivo de todo el sistema, pues esta tarea sería probablemente inabordable por muy distintas razones. Se pretende sentar las bases para que cada uno sea capaz de configurar su propio sistema en función de la tarjeta gráfica de la que disponga, de sus gustos y de sus preferencias. Al finalizar el taller, deberíamos ser capaces de instalar un entorno X, y de saber sacar partido a su increíble potencia.

Actividades

Debido a la complejidad del sistema X, se recomienda la lectura de los siguientes documentos para asentar conceptos antes de empezar a trabajar sobre el sistema. Además, éstos nos aportarán conocimientos suplementarios, que pueden ser puestos en práctica a lo largo del taller.

16. `http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html single/XWindow-Overview-HOWTO.html`
Se trata de un documento sencillo, que sirve para asimilar los conceptos básicos que conciernen al sistema.

17. `http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html single/XWindow-User-Howto.html`
Es un documento con contenidos más avanzados que el anterior, pero también recomendable.

11.2. Instalación del sistema básico

Como ya se ha dicho, el X Window System es un sistema muy complejo que integra muchísimas librerías y aplicaciones, algunas de las cuales son fundamentales para su funcionamiento, aunque la mayoría sólo deben ser instaladas si, por nuestras necesidades, precisamos de ellas. Ésta es una de las razones por las que, en Debian, el sistema viene distribuido en muchos paquetes distintos, de los que sólo instalaremos aquellos que sean necesarios.

11.2.1. Distintas estrategias para la instalación de los paquetes

Debido a que la interdependencia entre los distintos paquetes es muy fuerte, podemos aprovechar este hecho para que sea el mismo sistema de gestión de paquetes el que instale todos aquellos que considere necesarios para el correcto funcionamiento de una aplicación de alto nivel, entre los cuales se encontrarán, obviamente, todos los paquetes básicos del sistema. Así pues, podríamos utilizar `dselect` o `apt-get` para instalar una de estas aplicaciones. Pero ésta no es una buena estrategia, ya que implica la pérdida del control de los paquetes que estamos instalando y además puede implicar la omisión de algunos paquetes esenciales, de los que, por el motivo que sea, no se haya hecho referencia durante el cálculo de dependencias.

Por este motivo, se recomienda la construcción del sistema paso a paso, para ir comprendiendo qué paquetes se están instalando en cada momento, y el porqué.

ANOTACIONES

11.2.2. Instalación de paquetes básicos

Los paquetes básicos del sistema se pueden agrupar, de forma general, en:

core. Paquetes esenciales e imprescindibles para el correcto funcionamiento del sistema (`x-window-system-core`).

system. Paquetes esenciales cuya presencia es altamente recomendable (`x-window-system`).

fonts. Paquetes de fuentes, algunos de ellos imprescindibles para disponer, al menos, de un tipo de fuentes en el entorno.

utils. Paquetes de programas que correrán sobre el sistema gráfico. Son, en el fondo, los que dan la razón de existencia del sistema.

libs. Paquetes de librerías del entorno gráfico, las cuales proporcionan funciones a otros programas.

Podemos instalar los paquetes básicos del sistema mediante la orden siguiente:

```
brau:˜# apt-get install x-window-system
```

Con ello se calcularán las dependencias entre paquetes y se nos informará de que debe instalarse alrededor de una treintena de paquetes. Para proceder, se nos pide la confirmación, a la cual contestaremos afirmativamente.

Nota

Después de que se hayan desempaquetado los paquetes, se ejecutarán automáticamente los *scripts* de configuración de varios de ellos. Siempre es posible interrumpir la ejecución de estos *scripts* con la combinación de teclas Ctrl-C, y reiniciar el proceso volviendo a ejecutar el comando anterior.

ANOTACIONES

Durante la ejecución del *script* de configuración del paquete `X-server-common`, se nos preguntará si queremos que sea `debconf` quien configure los *wrappers*. La opción más sencilla y práctica es permitir que la configuración se haga de forma automática, y seguir adelante.

El *script* de configuración de `Xserver-xfree86` nos situará en la interfaz de configuración del servidor. Lo primero que se nos pregunta es si deseamos que se ejecute un test de autodetección del *driver* que debe cargarse para nuestra tarjeta. Es buena opción contestar afirmativamente a esta pregunta, ya que si el test falla o detecta más de una posibilidad, nos dará acceso a la pantalla de selección de *drivers*. Si debemos seleccionar el *driver* manualmente, en primer lugar hay que leer la documentación de la tarjeta gráfica; si no disponemos de ella, también podemos utilizar el comando `lspci` para informarnos de cómo ésta ha sido detectada por el *kernel* (generalmente la última línea de retorno). Una vez dispongamos de esta información, nos dirigiremos a `http://xfree86.org/current/Status.html` para informarnos de cuál es el *driver* que debemos cargar. Con esto nos situamos en la pantalla siguiente. En ésta se nos pedirá si deseamos que sea `debconf` quien nos guíe en la configuración del archivo `/etc/X11/XF86Config-4`, que es el archivo fundamental de configuración del servidor. No es mala idea contestar afirmativamente, y, al final, comprobar si la configuración resultante es válida o no. En primer lugar se nos pregunta si queremos usar el *kernel framebuffer device interface*, y se nos sugiere que lo hagamos; en principio, contestaremos afirmativamente, pero teniendo muy presente que esto es una fuente potencial de problemas, de manera que, una vez hayamos finalizado la configuración, si al intentar arrancar el servidor algo va mal, deberemos estar atentos al contenido de la línea, y probar si con el valor “`false`” funciona:

```
#/etc/X11/XF86Config-4
Section "Device"
    Identifier    "Generic Video Card"
    Option        " UseFBDev"          "true"
```

Lo siguiente que configura el mismo *script* es el teclado. Por lo general, y en especial sobre la plataforma PC, la sugerencia de utilizar el tipo `xfree86` debe funcionar correctamente, aunque en las panta-

ANOTACIONES

llas posteriores terminaremos de ajustar su funcionamiento. En la siguiente pantalla contestaremos qué tipo de teclado es el que tenemos; por lo general `pc105` debe funcionar correctamente.

Ha llegado el momento de seleccionar el idioma del teclado. En este campo entraremos `es`; en cuanto al campo de variación idiomática, puede entrarse el mismo valor anterior, pero es recomendable dejarlo en blanco, ya que si se entra el valor `es`, esta directiva también se convierte en una fuente potencial de errores. Si más adelante, y en el caso de que hayamos rellenado el campo de variación idiomática, al parar el servidor se visualiza, entre otras líneas, las siguientes:

```
The XKEYBOARD keymap compiler (xkbcomp) reports:
> Error:  No Symbols named "es" in the include file "es"
>         Exiting
>         Abandoning symbols file "default"
Errors from xkbcomp are not fatal to the X server
```

o si no podemos utilizar ciertos caracteres como puede ser "|", deberemos comentar la directiva de variante idiomática en el archivo principal de configuración:

```
#/etc/X11/XF86Config-4

Section " InputDevice"
              Identifier    "Generic Keyboard"

#             Option        "XkbVariant" "es"
```

El campo siguiente, referente a la personalización del teclado, se deja en blanco si no se tiene ningún propósito al respecto. Con esto habremos terminado con la configuración del teclado y pasaremos a la configuración del ratón. Para ello, en primer lugar, se nos pide en qué puerto está ubicado (actualmente, la mayoría de ellos en `/dev/psaux`, la primera opción). A continuación se nos pregunta de qué tipo de ratón disponemos (nuevamente, la primera opción, `PS/2`, suele ser la más habitual). Terminada la configuración del ratón, entramos en la configuración de la pantalla. En primer lugar se nos pregunta si disponemos de una pantalla tipo LCD. Acto seguido se nos pedirá en qué grado de dificultad queremos entrar los pará-

ANOTACIONES

metros referentes a nuestro monitor. La forma más simple (`Simple`) sólo nos pedirá el tamaño del monitor, mientras que en la más compleja (`Advanced`) se nos pedirá las frecuencias de refresco horizontales y verticales del monitor, entre otros. Seguidamente se nos preguntará con qué resoluciones queremos trabajar y, en función de los parámetros que hayamos entrado en el apartado anterior, se nos harán distintas sugerencias. A continuación se nos pedirá cuál será la resolución de color que se usará por defecto al arrancar el servidor (actualmente la mayoría de tarjetas soportan, sin ningún problema, resoluciones de 24 bits). Con esto habremos terminado la configuración del servidor.

Nota

Una buena forma de conseguir un fichero `/etc/X11/XF86Config-4`, que funcione, es copiar el que genera KNOPPIX cuando entramos en modo gráfico; así pues, podemos hacer una copia de este fichero, en un disquete, por ejemplo, y compararlo con el que hemos generado mediante el *script* de configuración del servidor.

Actividad

18. Una vez terminada la instalación, editar el fichero `/etc/X11/XF86Config-4`, e intentar asociar las distintas directrices a los parámetros que hemos entrado durante la ejecución del *script* de configuración del servidor.

11.2.3. Inicialización del servidor

Ha llegado el momento de comprobar si el fichero de configuración del servidor es correcto, y en consecuencia el servidor arranca como es debido. Para ello basta con ejecutar el comando `startx`.

Si todo funciona correctamente, al cabo de unos momentos nuestra pantalla adquirirá un fondo mallado de colores grisáceos, y en el centro de la pantalla aparecerá un aspa. Hemos dado un gran paso adelante, pues configurar el servidor para que arranque es lo más difícil del entorno X. Ahora sólo es cuestión de tiempo para terminar

ANOTACIONES

dando al entorno el aspecto que deseamos. Mediante el ratón podemos mover el aspa, y pulsando los botones del medio e izquierdo, podemos explorar un poco las posibilidades de este entorno gráfico un tanto rudimentario. Para salir de él y continuar con la configuración del sistema, hay que pulsar el botón izquierdo del ratón, situarnos sobre `Exit` opción `Yes, really quit`, o simplemente pulsar la combinación de teclas CTRL-ALT-BackSpace.

Si por el contrario, al cabo de unos momentos retornamos a la consola alfanumérica, es que el servidor no ha podido arrancar adecuadamente. Ha llegado el momento de estudiar detenidamente el fichero de log (`/var/log/XFree86.0.log`) e intentar detectar las posibles fuentes de errores. Las más comunes suelen ser: mala elección del *driver* que hay que cargar (si el proceso de selección lo hemos dejado en manos del *script*, deberemos consultar la página antes mencionada para asegurarnos de que el *driver* que el *script* ha escogido es el correcto), poner la directiva `UseFBDev` a `true`, cuando debe estar en `false`, usar resoluciones o frecuencias de refresco más altas de las que la tarjeta puede soportar, etc.

Otra posibilidad es que, aun estando soportada la tarjeta por el Xfree86 Project, puede que el paquete que estemos usando no la soporte. Frente a esta situación, se sugiere que se incorpore la siguiente línea en el fichero `/etc/apt/sources.list`:

```
deb http://people.debian.org/~blade/woody/i386/ ./
```

y que, tras ejecutar el comando `apt-get update`, se vuelva ejecutar el comando de instalación del X-Window-System (`apt-get install X-Window-System`). Con esto acudiremos a una página no oficial para obtener los paquetes de versiones más actualizadas del sistema X Window, y en ellos, probablemente, ya se dé soporte a nuestra tarjeta.

Con todo esto, hay que destacar que la correcta configuración de una tarjeta de vídeo puede llegar a ser una tarea verdaderamente difícil. Para ello, acudiremos a buscadores donde entraremos palabras clave referentes a nuestra tarjeta junto con "X" o "Linux" para ver si encontramos documentación al respecto.

ANOTACIONES

Llegados a este punto, mediante el comando siguiente evitaremos que cada vez que arranquemos el ordenador entremos en el modo gráfico:

```
brau:˜# rm /etc/rc2.d/S99xdm
```

Esto nos será útil mientras no tengamos configurado totalmente este entorno. Una vez terminado, ya será cada usuario el que decidirá si quiere o no, que al arrancar el ordenador se entre en el modo gráfico. Para hacerlo, sólo habrá que volver a crear el enlace simbólico que hemos borrado con el comando anterior:

```
brau:˜# cd /etc/rc2.d
brau:/etc/rc2.d# ln -s ../init.d/xdm S99xdm
```

11.2.4. El fichero de log

Aunque el arranque del servidor haya resultado exitoso, no por ello debemos dejar de analizar el contendido del fichero principal de log del sistema X, `/var/log/XFree86.0.log`, ya que esto nos ayudará a depurar fallos y errores menores que no han interrumpido el arranque del servidor, pero sí que hacen bajar su rendimiento.

Algunos ejemplos típicos pueden ser: eliminar la línea del fichero de configuración que hace referencia a las fuentes cirílicas, si nunca las vamos a utilizar, ya que ni tan siquiera las hemos instalado; eliminar de este mismo fichero todo lo que hace referencia al Generic Mouse, pues lo hace incompatible con el que hemos configurado nosotros, etc.

11.2.5. El servidor de fuentes

El sistema gráfico en sí no lleva asociado ningún tipo de base de datos acerca de las fuentes disponibles, pues esto no entra en el dominio de sus funciones. Ésta es la tarea del servidor de fuentes xfs (`/usr/bin/X11/xfs`), así, cualquier programa cliente podrá usar las fuentes disponibles, que se arrancan automáticamente al lanzar el servidor X en forma de deamon. Existen multitud de paquetes para agregar más fuentes al conjunto que hemos instalado por defecto.

Una aplicación que puede ser útil para conocer qué fuentes tenemos instaladas y qué aspecto tienen es `xfontsel`. Se trata de una aplicación escrita mediante librerías de athena (como el resto de los xbase-clients, grupo al cual pertenece `xfontsel`), en la cual, mediante menús desplegables se seleccionan las opciones disponibles de configuración de las fuentes y nos muestra algunos caracteres de ejemplo. Para conocer los caracteres de una fuente se puede usar la aplicación `xfd`.

Así pues, una vez sepamos qué fuente queremos utilizar para un determinado programa, podemos lanzarlo desde una xterm con el argumento "-fn" seguido de la descripción de la fuente (otros programas pueden usar nombres distintos para este argumento). Un ejemplo puede ser:

```
xterm -fn -misc-fixed-medium-r-normal--20-140-100-100-c-100-iso8859-1 &
```

11.3. *Window managers*

Los *window managers* son programas clientes (en realidad se los llama meta-clientes) encargados de gestionar las distintas ventanas que corren sobre el entorno gráfico y de su presentación, así como de lanzar otros clientes (aplicaciones). A estas alturas, ya tenemos un *window manager* instalado, el twm, ya que una instalación completa de un sistema X requiere como mínimo de un *window manager*, aunque éste no forme parte del servidor, que corra sobre él. Como ya hemos visto, el twm es muy rudimentario, puede ser, pues, que nos interese instalar algunos más complejos, como puede ser WindowMaker, BlackBox, qvwm, etc. Vamos pues a instalar algunos de ellos y a probarlos. El objetivo final, siguiendo la filosofía GNU, es que cada usuario termine usando el software que prefiera. Así pues, se deja que, una vez conocidos algunos *window managers* existentes, sea el propio lector quien decida cuál va a usar. Obviamente, es posible tener más de un *window manager* instalado, aunque sólo se pueda hacer correr uno por sesión. (Sin embargo, si que podemos, como ejemplo de la flexibilidad del sistema X, tener dos *window manager* corriendo en un mismo ordenador, cada uno en un terminal distinto).

ANOTACIONES

En primer lugar instalaremos, a modo de ejemplo, el qvwm. Se trata, como veremos al lanzarlo, de un *window manager* que simula un entorno que probablemente nos sea conocido. Para hacerlo, basta con ejecutar la instrucción:

```
brau:~# apt-get install qvwm
```

En este momento ya tenemos dos *window managers* instalados: el twm y el qvwm. Para hacer correr uno u otro bastará con indicar la ruta completa de ubicación del *window manager* que deseamos utilizar después del comando `startx` (recordemos que el comando `whereis` nos puede ser muy útil a la hora de buscar la ubicación de un fichero). Así pues, para hacer correr el *window manager* que acabamos de instalar, bastará con ejecutar:

```
brau:~# startx /usr/bin/X11/qvwm
```

Para utilizar el twm se debería haber especificado su ruta completa de la forma siguiente:

```
brau:~# startx /usr/bin/X11/twm
```

Tener que especificar cada vez qué *window manager* deseamos utilizar, una vez nos hayamos decidido por uno en concreto, puede ser un tanto pesado. Para especificar qué *window manager* debe usarse en caso de que después del comando `startx` no se especifique ninguno en concreto, crearemos el archivo `.xsession` en el directorio raíz del usuario con el contenido siguiente, en el caso que quisiéramos que el *window manager* por defecto fuese el twm, por ejemplo:

```
#~/.xsession
exec twm
```

Si quisiéramos que fuese qvwm el*window manager* por defecto, bastaría con cambiar `twm` por `qvwm`. La ejecución de los distintos procesos que implican el arranque del entorno gráfico, así como los ficheros de configuración que se van leyendo durante este proceso, están fuertemente determinados. Así pues, creando el fichero anterior, lo que hemos hecho es editar uno de los últimos ficheros (los

ANOTACIONES

que residen en el directorio raíz del usuario) que se leen antes de entrar en el entorno gráfico. Este fichero, por tanto, nos permite modificar algunos aspectos de los que se han determinado por defecto dentro del sistema y que están definidos en los ficheros residentes en `/etc/X11` y sus subdirectorios.

Para finalizar esta sección instalaremos un *window manager* muy utilizado en el mundo GNU/Linux, que se caracteriza por su versatilidad y su escaso consumo de recursos: el WindowMaker:

```
brau:˜# apt-get install wmaker
```

Lectura complementaria

Animamos al lector a familiarizarse un poco con este *window manager* y a que visite su página para ampliar conocimientos:
http://www.windowmaker.org/.

Hemos instalado ya tres *window managers*, y seguramente instalaremos más. Aparte del método anteriormente descrito para preestablecer cuál queremos ejecutar por defecto, podemos utilizar el menú del comando `update-alternatives` para establecerlo:

```
brau:˜# update-alternatives x-window-manager
```

11.4. X *Session manager*

Los *session managers* son programas que pueden hacerse correr sobre una sesión gráfica y que nos permitirán establecer y modificar parámetros de ésta. xms es el *session manager* que viene por defecto con la instalación que hemos hecho del servidor gráfico. Podemos lanzarlo desde un terminal X (para lanzar una xterm, pulsaremos el botón del medio del ratón y seleccionaremos Programs/Xshells/Xterm) , mediante el comando `xsm`.

Una vez lanzado xsm mediante Checkpoint, podemos guardar la configuración de la sesión actual (esencialmente referente a las aplicaciones que tengamos corriendo), gestionar los procesos que están corriendo mediante Client List, consultar el log de la sesión o cerrar la sesión guardando la configuración actual.

Aparte de xsm, existen otros *session managers*. Éstos acostumbran a ser una parte más de los *desktop managers*, y están tan integrados en ellos que a veces resulta difícil reconocer sus acciones. Un ejemplo

típico es la pregunta que se nos formula acerca de si deseamos guardar la sesión al cerrar KDE.

11.5. X *Display manager*

Al finalizar la instalación del Xserver, sugeríamos que se eliminara el enlace simbólico `/etc/rc2.d/S99xdm` para evitar que al volver a arrancar el sistema al entrar al *runlevel* 2 se ejecutase `xdm`, acrónimo de X *display manager*. Éste es el *display manager* que el paquete `X-Window-System` instala por defecto. Los *display managers* son los programas encargados de gestionar quién, desde dónde, y cómo puede entrar un usuario al entorno gráfico. Para lanzarlo, lo haremos como con cualquier otro daemon:

```
brau:˜# /etc/init.d/xdm start
```

Para pararlo, también utilizaremos el mismo procedimiento que seguiríamos para detener cualquier otro daemon, con la salvedad de que debemos pulsar la combinación de teclas Ctrl+Alt+F1, para salir del entorno gráfico y situarnos en la tty1, por ejemplo, en vez de utilizar la combinación que se usa para cambiar de ttys en entornos alfanuméricos:

```
brau:˜# /etc/init.d/xdm stop
```

Como hemos comprobado, el *display manager* nos pide un *login* y un *password*, los mismos que utilizamos para acceder al sistema por las ttys, si no es que hemos impuesto alguna restricción. Tras validarnos, entramos en el modo gráfico del mismo modo como lo hacíamos mediante el comando `startx`. La diferencia radica en que, cuando terminemos la sesión gráfica, el servidor no se parará, si no que continúa corriendo el `xdm`.

Uno de los inconvenientes de xdm es que no nos permite seleccionar con qué *window manager* queremos trabajar. Pero existen otros *display managers*, como pueden ser wdm (de WindowMaker), gmd (del proyecto GNOME), o kdm (del proyecto KDE), que sí que lo permiten.

ANOTACIONES

Podemos instalar el wdm, para ver su aspecto, y para conocer otro *display manager*:

```
brau:˜# apt-get install wdm
```

Al ejecutarse el *script* de postinstalación, se nos preguntará qué *display manager* queremos usar, xdm, que ya teníamos instalado, o wdm. Seleccionaremos este último, para que se cree el enlace necesario para que se lanze wdm com *display manager* durante el proceso de arranque del sistema (si existe el fichero `/etc/rc2.d/S99xdm`, es mejor borrarlo para evitar mensajes de *warning* al arrancar el sistema). Si no queremos que se arranque automáticamente ningún *display manager* al arrancar el sistema, bastará con eliminar los enlaces necesarios, es decir, el fichero `/etc/rc2.d/wdm`. El archivo `/etc/X11/default-display-manager` marca el *display manager* que se usa por defecto.

Una vez arrancada una sesión X desde el *display manager*, es decir, una vez hayamos lanzado el pertinente *window maker*, puede ser interesante ejecutar el comando `pstree` para ver las relaciones de dependencia entre los distintos procesos que están corriendo en este momento, junto con la información que nos aportará la línea `ps aux`.

11.6. *Desktop managers*

La aparición de distintas *toolkits*, así como el desarrollo de diversos proyectos que desarrollaban o usaban librerías del entorno gráfico, hizo aparecer proyectos que buscasen la unificación de todos estos esfuerzos. Fue entonces cuando apareció un concepto nuevo en el entorno X: el de *desktop manager*. Los *desktop managers* son proyectos que pretenden sentar las bases para la unificación y estandarización, tanto de presentación como de políticas de programación y de desarrollo de aplicaciones. Uno de los primeros en aparecer fue el CDE (Common Desktop Manager), aunque actualmente los dos proyectos más destacados en este sentido son: GNOME y KDE, a los cuales, dado su alto grado de implementación y de desarrollo, dedicaremos una subsección respectivamente. Pero antes podemos nombrar otros *desktop managers*, como pueden ser: GNUStep, ROX, GTK+Xfce o UDE.

ANOTACIONES

11.6.1. GNOME

GNOME es un proyecto que forma parte de GNU, que se caracteriza por no necesitar estrictamente de un *window manager* en concreto, aunque se recomienda que se use alguno que garantice su correcto funcionamiento (a GNOME-*compliant window manager*) como pueden ser: IceWM o Sawfish. Aun así, para respetar las preferencias y la libertad del usuario, GNOME, en su control panel dispone siempre de un *window manager* selector, que nos permite escoger qué *window manager* queremos usar. GNOME está basado en Gtk *toolkit*, las propias librerías desarrolladas dentro del proyecto, conocidas como *gnome-libs específiques*.

Como todos los *desktop managers*, GNOME dispone de su propio panel, de su gestor de archivos Nautilus y de su control panel: GNOME Control Panel.

Para hacer una instalación básica de GNOME, instalaremos el paquete siguiente junto con todas sus dependencias:

```
brau:˜# apt-get install gnome-session
```

Tal como se ha dicho, aunque GNOME no exija el uso de ningún *window manager* determinado, se recomienda que éste sea GNOME-*compliant window manager*. Vamos pues a instalar Sawfish, que fue desarrollado estrictamente para cumplir este requisito. Instalemos el paquete y todas sus dependencias:

```
brau:˜# apt-get install sawfish-gnome
```

Tenemos, pues, otro *window maker* instalado. Detendremos el *display manager* y volveremos a lanzarlo para que este nuevo *window maker* se integre en él (GNOME también tiene su propio *display manager*, gdm, que podemos instalar si lo deseamos). Ahora tenemos dos posibilidades para conseguir nuestro objetivo: hacer correr GNOME. La primera es arrancar Sawfish, desde el *display manager* o mediante `startx` y, una vez dentro, lanzar `gnome-session` desde un terminal X, o bien operar de modo inverso, es decir, arrancar GNOME por los mismos procedimientos que Sawfish, y luego lanzar `sawfish` desde un terminal X. Se recomienda proceder de

ANOTACIONES

la última forma si queremos que la próxima vez que arranquemos GNOME se ejecute Swafish (será el propio *sesión manager* de GNOME el encargado de realizar y registrar los cambios necesarios para que esto ocurra).

Una vez familiarizados un poco con GNOME, lo que podemos hacer es instalar algunos paquetes que nos pueden ser útiles, en concreto `gnome-help` y `gnome-terminal`, el primero nos ofrece una interfaz donde podremos leer *mans*, ficheros de texto en un entorno grafico, y el segundo instala el xterm propio de GNOME.

Una aplicación que dará un aspecto mucho más amigable a GNOME será Nautilus, concebido, en principio, como un gestor de ficheros, pero que además nos permitirá configurar el *desktop*. Para su correcto funcionamiento, Nautilus necesita de Mozilla M18 y de GNOME Helix. Por tanto, los paquetes que habrá que instalar serán los siguientes: `nautilus-suggested`, para hacer la instalación de Nautilus y `mozilla`, para hacer una instalación completa del navegador (en cuanto a su configuración, podemos instalar las fuentes True Type, y podemos poner en `auto` el sound daemon's dsp wrapper).

GNOME es un proyecto en constante desarrollo y que ofrece multitud de aplicaciones. Por eso, una vez instalado su sistema básico, sugerimos al lector que visite su página `http://www.gnome.org` y que sea él mismo el que explore, descubra y decida cuál de ellas pueden ser de su interés para instalárselas. La sola ejecución de la línea siguiente nos puede dar una idea de la magnitud del proyecto:

```
brau:˜# apt-cache search gnome
```

Debian Woody integra la versión 1.4 de GNOME. Estando ya disponible la versión 2.2, proponemos modificar el contenido de `/etc/apt/sources.list` añadiendo las líneas siguientes, para poder actualizar o instalar de raíz los paquetes de esta versión "debianizados":

```
#GNOME2
deb http://ftp.acc.umu.se/mirror/mirrors.evilgeniuses.org.uk/debian/backports/woody gnome2.2/
```

ANOTACIONES

```
deb-src http://ftp.acc.umu.se/mirror/mirrors.evilgeniuses.org.uk/debian/backports/woody gnome2.2/
```

Una vez hayamos hecho esto, bastará con ejecutar los siguientes comandos para disponer de dichas actualizaciones:

```
apt-get update apt-get install gnome
```

11.6.2. KDE

KDE, a diferencia de GNOME, sí que necesita de un *window manager* concreto, se trata de kwm, basado en Qt *toolkit* y en las propias kdelibs. También dispone de su *launcher panel*: kpanel, de su propio gestor de archivos: Konquest y de su utilidad de configuración: Control Panel. Obviamente, KDE puede estar instalado en el mismo sistema donde hayamos instalado GNOME, e incluso hay aplicaciones pertenecientes a un *desktop manager* que pueden correr en el otro. Además, KDE también tiene su propio *display manager* (kdm) junto con muchísimas más aplicaciones. Nuevamente se recomienda al lector una visita a su página para conocer sus posibilidades: `http://www.kde.org`, y la página del encargado de integrar KDE en Debian `http://people.debian.org/ ccheney/`. Asimismo, podemos ejecutar la línea siguiente para ver la integración de KDE en Debain:

```
brau:˜# apt-cache search kde
```

Los paquetes básicos de KDE están en el paquete `kdebase`. Éste será, pues, el primero que instalaremos:

```
brau:˜# apt-get install kdebase
```

Nuevamente deberemos reinicializar nuestro *window manager* para tener acceso al *desktop manager* recién instalado. Una vez hecho esto, podemos proceder a instalar el gestor de archivos, paquete `konquest`. Mediante el paquete `kde-i18-es` podemos instalar los ficheros necesarios para que KDE trabaje en castellano.

ANOTACIONES

A partir de este punto, ya será cada usuario el que instalará los distintos paquetes del proyecto que le sean de interés.

Por lo que a versiones se refiere, Debian Woody integra la versión 2. Para disponer de versiones más recientes, hay que recurrir a paquetes no oficiales. Si se desea tener acceso a estos paquetes, podemos añadir la línea siguiente a `/etc/apt/sources.list` y actualizar la base de datos mediante el comando `apt-get update`:

```
#KDE3
deb http://people.debian.org/˜schoepf/kde3/woody ./
```

Tal como ya hacíamos, para preestablecer el *window maker* por defecto, utilizaremos el menú del comando `update-alternatives` para seleccionar el *session manager:*

```
brau:˜# update-alternatives x-session-manager
```

11.7. Personalización de algunos aspectos

El diseño del entorno gráfico X responde al ambicioso objetivo de sacar el máximo rendimiento del hardware disponible, usando el mínimo de sus recursos, y ofrecer la máxima flexibilidad posible. La estructura de servidor cliente en que se basa este sistema hace posible que estos objetivos sean una realidad, y las aparentes dificultades con las que se encuentra el usuario novel desaparecen rápidamente con un poco de práctica para permitir que afloren las múltiples ventajas que ofrece este diseño. En esta sección sólo se pretende dar una somera idea de la potencia de este sistema, la cual se pone plenamente de manifiesto cuando se trabaja en red, aunque queda un poco ensombrecida cuando se trabaja en un sistema *stand alone* como sobre el que se desarrolla todo este curso. De todas formas se introducen algunos conceptos que pueden ser de utilidad cuando se entra a trabajar en red.

11.7.1. Personalización de aspectos locales

En general los archivos de configuración se hallan en el directorio `/etc/X11/` o en algunos de sus subdirectorios. De forma personali-

ANOTACIONES

zada, cada usuario puede redefinir los parámetros de configuración y añadir nuevos creando o editando en su directorio de *home* los ficheros que llevan el mismo nombre que los de configuración general pero precedidos de un ".". Se podrán redefinir o establecer todos aquellos parámetros que no requieran de permisos de superusuario, ya que los archivos de *home* se procesan después de los de configuración general, y los parámetros siempre tomarán el último valor que se les asigne.

Xsession

`/etc/X11/Xsession` es un *script* que se ejecuta al entrar en una sesión de usuario.

Este *script* es el que gobierna todo el proceso de arranque de la sesión hasta que podemos empezar a trabajar, y también es el encargado de gestionar los mensajes de errores que se puedan producir durante este proceso, los cuales se registran en `$HOME/.xsession-errors`.

En `$HOME/.xsession` es donde personalizaremos el arranque para un usuario en particular. Así pues, si deseamos que el *window manager* sea *blackbox*, y que se arranque automáticamente bbkeys en *background* al iniciar la sesión, éste contendrá las siguientes líneas:

```
bbkeys &
blackbox
```

Xresources

En el archivo `$HOME/.Xresources` personalizaremos el aspecto de las distintas aplicaciones. La sintaxis es *application*parameter: value*. Así pues, si quisiéramos invertir los colores de la aplicación xterm, añadiríamos la línea siguiente en el fichero:

```
Xterm*reverseVideo: true
```

ANOTACIONES

El comando `xrdb` es el encargado de gestionar la base de datos de Xresources. Mediante "`xrdb -query`" podemos conocer todas las propiedades establecidas y su valor, y mediante el parámetro "`-display`" obtendremos un listado de todos los parámetros que acepta el comando. Si a éste le pasamos como parámetro la ubicación de un fichero, leerá de él todas las definiciones de parámetros.

Xmodmap

El servidor gráfico usa la tabla de códigos de caracteres para hacer la conversión de señales provenientes del teclado (*server-independent*) a símbolos del sistema (*server-dependent*).

La tabla de conversión que hay que usar ha sido seleccionada durante el proceso de configuración del teclado, pero el comando `xmodmap` nos permite modificar su contenido. Un ejemplo de su uso puede ser el siguiente:

```
brau:~# xmosmap -e "keycode 127 = Delete"
brau:~# xmosmap -e "keycode 22 = BackSpace"
```

Nota

Estos comandos fueron utilizados durante mucho tiempo debido a la inversión de asignación de símbolos en las tablas de conversión; actualmente, sin embargo, este problema está resuelto.

Mediante los parámetros "`-pk`" `xmodmap` nos devolverá todo el contenido de la tabla de conversión que se está usando.

11.7.2. Personalización de aspectos de red

Los aspectos aquí presentados también son de interés para un sistema *stand alone*, ya que, como todo el sistema operativo, el sistema X usa siempre un diseño orientado a red.

$DISPLAY

La variable `$DISPLAY` sirve para indicar al cliente con qué servidor debe comunicarse. Su sintaxis es la siguiente: *hostname:display number.screen number*. Así pues, si hubiésemos definido otro terminal

ANOTACIONES

gráfico al sistema X, añadiendo la línea siguiente a `/etc/X11/xdm/Xservers`:

```
:1 local /usr/X11R6/bin/X vt8
```

podríamos lanzar una aplicación gráfica desde un xterm de un terminal gráfico a otro definiendo la variable adecuadamente. Por todo lo cual, si quisiéramos lanzar `xeyes` desde el primer terminal gráfico, vía xterm, y "displayarlo" en el segundo, procederíamos de la forma siguiente:

```
brau:~$ set DISPLAY :0.1; export DISPLAY
brau:~$ xeyes &
```

Si entramos en una sesión gráfica, abrimos un xterm, cambiamos de usuario mediante el comando `su` y probamos de lanzar una aplicación gráfica se nos devolverá un mensaje de error indicándonos que se puede establecer conexión con el servidor. Una estrategia para evitar este problema es utilizar con el parámetro "–`p`" para que se exporte todo el conjunto de variables de entorno, y evitar así que el servidor rechace nuestra petición de conexión. Esta práctica puede ser muy útil para lanzar programas de configuración que necesitan permisos de *root*, ya que nos evitará tener que entrar en el entorno gráfico como *root* (práctica no muy recomendable, y que, aunque por defecto se permita, en muchas ocasiones se restringe manualmente).

`xhost` y `xauth`

El comando `xhost` permite establecer qué equipos pueden acceder al servidor gráfico de forma remota, es decir, qué máquinas cliente pueden lanzar una aplicación para ser "displayada" en el servidor. Su sintaxis es la siguiente: `xhost +`*hostname*. Si no se especifica ningún *hostname*, cualquier máquina podrá lanzar aplicaciones sobre el servidor. Por defecto, no se permite la conexión desde ningún equipo remoto.

El comando `xauth` sirve para determinar qué usuarios pueden lanzar aplicaciones sobre el servidor gráfico. Así pues, mediante la combinación de estos dos comandos podremos establecer una política

ANOTACIONES

de seguridad de acceso al servidor X bastante razonable. xhost + para los *stand alone*.

11.8. Configuración de impresoras

La tarea de configurar impresoras se puede ver facilitada a partir del entorno gráfico. Existen multitud de aplicaciones para configurar el sistema de impresión nativo (basado en lpd como servidor y lpr como cliente), y otras que sustituyen este sistema por uno propio, comúnmente también basado en la estructura cliente servidor. En esta sección presentaremos dos de estas herramientas: CUPS y LPRng, ambos con su sistema propio de servidor de impresión.

Para la instalación de LPRng hay que instalar necesariamente el paquete `lprng`. El paquete que contiene la GUI es de configuración del fichero `/etc/printcap` basado en LPRng es `lprngtool`. Obviamente, para hacer modificaciones en dicho fichero de configuración, habrá que ejecutar el comando `lprngtool` con privilegios de *root*. Adicionalmente, se recomienda instalar el paquete `printop`, que aporta una interfaz gráfica para todo el sistema de impresión, y el paquete `lprng-doc`, el cual contiene toda la documentación de este sistema de impresión.

Para instalar CUPS (Common Linux Printing System) habrá que instalar el paquete del servidor de impresión, `cupsys`; y se recomienda instalar, junto a este paquete, el de clientes de impresión, paquete `cupsys-client`. También se puede instalar el paquete `cupsys-bsd` para disponer de los comandos habituales en el sistema de impresión de BSD. Por lo que se refiere a *drivers*, los paquetes `cupsys-driver-gimpprint` y `cupsomatic-ppd`, aportan gran número de ellos junto con numerosos algoritmos de procesamiento de imagen, definición de tamaños de papel, etc.

11.9. OpenOffice

En esta sección, aunque se aparte un poco del objeto principal del curso, presentaremos una *suite* ofimática que puede ser de

ANOTACIONES

extraordinaria utilidad para aquellos que estén acostumbrados a utilizar software de este tipo. Se trata de OpenOffice.org (http://www.openoffice.org), proyecto derivado de StarOffice, de Sun MicroSystems. Hay que destacar que este proyecto es multiplataforma, y que por tanto puede ser implementado en otros sistemas operativos no tipo UNIX.

Si bien esta *suite* no está integrada en Debian, podemos recurrir, como ya hemos hecho con anterioridad, a fuentes no oficiales para obtener sus paquetes:

```
#OpenOffice
deb http://ftp.freenet.de/pub/ftp.vpn-junkies.de/openoffice/ woody main contrib
```

Una vez hayamos añadido la línea anterior a `/etc/apt/sources.list` y hayamos actualizado la base de datos mediante `apt-get update`, bastará con ejecutar la línea siguiente para proceder a la instalación:

```
brau:˜# apt-get install openoffice.org openoffice.org-l10n-en
```

Sólo hay que tener presente, durante el proceso de instalación (el cual debe ejecutarse bajo un entorno gráfico) que hay que hacer la instalación para red. Así, sólo será necesario que en el directorio *home* de cada usuario exista un pequeño directorio donde se guardarán sus configuraciones personales.

Una vez se haya instalado el programa la primera vez, cada usuario deberá ejecutar el programa siguiente para que se cree su directorio:

```
/usr/lib/openoffice/program/setup
```

Una vez hecho esto y se hayan repuesto algunas preguntas, mediante el comando `openoffice` abriremos la suite.

ANOTACIONES

11.10. Conclusión

Con este taller se termina el material didáctico del segundo módulo del máster. En él hemos aprendido a instalar el entorno gráfico al sistema, el cual, como ya vimos en su momento, no es una parte fundamental dentro del sistema operativo. Pero está clara su utilidad en muchos casos, y ha sido en este punto cuando muchos afirman que el sistema operativo que ha sido objeto de estudio en este módulo puede ser una alternativa seria a otros sistemas. Por todo lo cual, los autores querríamos manifestar nuestro absoluto convencimiento de que GNU/Linux es un sistema operativo extraordinario, no sólo debido a su entorno gráfico, que es lo que quizás sorprenda más a primera vista, sino por un sinfín de argumentos, de entre los cuales podemos destacar su filosofía, robustez, adaptabilidad, potencia, niveles potenciales de seguridad, etc. Estamos convencidos de que este sistema operativo es una apuesta de futuro para la cual, si bien ya ha demostrado que es capaz de abrirse un espacio en el mundo de los sistemas operativos, sólo cabe esperar sorpresas positivas.

Deseamos haber proporcionado los conocimientos suficientes y transmitido el entusiasmo necesario para que el lector inicie aquí su propio camino en el mundo del software libre, y haber sabido abrir las puertas a una comunidad donde todo el mundo es bienvenido y respetado.

A. Tablas de comandos

A.1. Sistema de ficheros

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
ls [pattern]	Lista los contenidos de un directorio determinado Þ "-a" muestra todos los contenidos, incluso los que empiezan por "." Þ "-l" muestra la información completa relativa a los contenidos Þ "-h" acompañada de "-l" muestra el tamaño de los archivos en unidades de bytes, KB, MB, GB Þ "-S" ordena el listado por tamaño Þ "-w" muestra el listado por columnas Þ "-R" muestra el listado de forma recursiva, pasando por todos los subdirectorios Þ "--color" colorea el texto del listado según el tipo de archivo
cd [ruta]	Cambia al directorio especificado. Si no se especifica ninguno, va al *home* del usuario
pwd	Muestra la ruta completa hasta el directorio actual
find [parámetros] [path]	Busca un determinado fichero o directorio Þ "-iname pattern" busca recursiva del *pattern* a partir del directorio actual Þ "-ilname pattern" como "-iname" pero haciendo la búsqueda insensible a mayúsculas o minúsculas Þ "-maxdepth numLevels" realiza la búsqueda hasta el nivel de profundidad especificado Þ "-uid UID" fuerza la coincidencia de UID Þ "-gid GID" fuerza la coincidencia de GID
ln ruta [nombreNuevoLink]	Crea un *link* a un fichero o directorio. Si no se trata de un *link* simbólico, se hace una copia exacta y se modifica frente a cualquier actualización (*hard link*) Þ "-s" crea un enlace simbólico

A.2. Ayuda del sistema

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
man [numManual] comando	Muestra el contenido del manual del comando mediante un paginador
info comando	Otra fuente de ayuda, que a veces complementa los contenidos del *man*, y en otras es la única documentación que se mantiene
apropos palabraClave	Busca todos los comandos que contienen la palabra clave en la descripción de su manual

A.3. Permisos de los ficheros

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
chmod modo fichero	Establece los permisos de ficheros y/o directorios Þ "-R" el cambio se realiza de forma recursiva, es decir, se realiza sobre todos los ficheros y subdirectorios del directorio especificado
chown propietario[.grupo] fichero	Establece el propietario y el grupo (si éste se especifica) de ficheros y/o directorios Þ "-R" el comando actúa de forma recursiva
chgrp grupo fichero	Establece el grupo al que pertenecen ficheros y/o directorios Þ "-R" el comando actúa de forma recursiva
umask modo	Inicializa los permisos de todos los ficheros creados a partir del momento en que se ejecuta el comando

A.4. Copia y borrado de ficheros

COMANDO	SIGNIFICADO Y PARÁMETROS UTILES
rm fichero	Comando para borrar ficheros y/o directorios Þ "-f" suprime los mensajes de confirmación Þ "-R" el comando actúa de forma recursiva Þ "-i" pregunta antes de proceder al borrado
rmdir directori	Borra directorios vacíos
cp ficheroOrigen ficheroDestino	Copia archivos del origen al destino especificados Þ "-f" no pide confirmación Þ "-i" pide confirmación antes de copiar cada fichero Þ "-l" realiza enlaces en vez de copiar Þ "-R" actúa recursivamente copiando los contenidos de los subdirectorios Þ "-s" realiza enlaces simbólicos en vez de copiar
mv ficheroOrigen ficheroDestino	Mueve archivos del origen al destino especificados Þ "-f" no pide confirmación Þ "-i" pide confirmación antes de mover cada fichero

A.5. Parada o reinicio

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
logout	Sale del *shell* y devuelve al *login*
halt	Inicia el proceso de paro del sistema (nunca debemos apagar el ordenador hasta que no se "displaye" Power Down)
reboot	Reinicia el sistema, es decir, procede al paro de éste y luego "resetea" el ordenador

A.6. Operaciones con ficheros

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
cat [archivo]	Muestra el contenido del archivo o, si éste no se ha especificado, muestra por pantalla todo lo que se entra por teclado Þ "-n" enumera las líneas mostradas
less archivo	Paginador para mostrar contenido de archivos. Las opciones que se especifican a continuación NO son parámetros del comando, sino teclas de acceso directo a utilizar en su uso Þ "/pattern" dentro de "less" podemos realizar una búsqueda del *pattern* especificado precedido de "/"; Para continuar la búsqueda hacia delante o hacia atrás, usaremos las teclas "n" y "N" respectivamente Þ "FIN" va al final del archivo Þ "INICIO" va al principio del archivo Þ "AVPAG" avanza página Þ "RETURN" avanza una línea Þ "SPACE" avanza una página
more archivo	Muestra el contenido del archivo de una forma similar a "less" pero más simple (no permite, por ejemplo, la opción de retroceder)
grep [pattern] archivo	Busca las líneas que cumplen con el *pattern* y la muestra por pantalla Þ "-b" muestra el número de byte de cada línea encontrada Þ "-c" no muestra las líneas por pantalla, sino que las cuenta y devuelve el número Þ "-e pattern" útil cuando el *pattern* empieza por el carácter "-" Þ "-i" ignora la diferencia entre mayúsculas y minúsculas Þ "-n" muestra el número de línea Þ "-v" invierte el resultado de la búsqueda Þ "-w" selecciona sólo aquellas líneas donde el *pattern* es toda una palabra
cut archivo	Muestra los campos del archivo Þ "-d carácter" indica qué carácter es el delimitador de campo Þ "-f numCampo" indica qué campo debe mostrado Þ "-b listaBytes" del campo seleccionado, sólo se quiere ver los bytes indicados, separados por comas
wc fichero	Muestra el número de bytes, líneas y palabras del fichero Þ "-c" sólo muestra los bytes Þ "-l" sólo muestra el número de líneas Þ "-w" sólo muestra el número de palabras
diff archivo1 archivo2	Muestra les diferencias entre archivo1 y archivo2 Þ "-B" ignora líneas en blanco Þ "-i" ignora la diferencia entre mayúsculas y minúsculas Þ "-w" ignora los espacios en blanco

A.7. Compresión de ficheros y copias de seguridad

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
tar opciones archivo [archivoOrigen]	Une distintos archivos y directorios en uno solo (conocido como *tarball*, los cuales se caracterizan por terminar con ".tar") Þ "-cf" crea un archivo a partir de los archivos y directorios origen Þ "-cvf" como en el caso anterior, pero mostrando por pantalla el proceso Þ "-cvzf" como en el caso anterior, pero comprimiendo los archivos con gzip Þ "-x" para extraer el contenido de un *tarball* Þ "-xvzf" para extraer y descomprimir mostrando por pantalla los resultados del proceso (los archivos comprimidos se caracterizan por terminar en ".tar.gz" o ".tgz"
cpio	Para hacer copias de seguridad
gzip archivo	comprime el archivo y le da la extensión ".gz" Þ "-d" descomprime
gunzip archivo	Descomprime el archivo
zip archivoFinal archivoOrigen	Comprime todos los archivos origen en archivo final (el cual se caracteriza por terminar con ".zip" Þ "-e" encripta el archivo mediante una palabra clave que se pide al ejecutar el comando
unzip archivo	Descomprime el archivo

A.8. Operaciones de disco

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
df	Muestra el estado de las particiones montadas por lo que se refiere a espacio total, ocupado y libre Þ "-h" muestra la información en unidades de bytes, KB, MB, GB Þ "-m" muestra la información en unidades de MB
du [archivo]	Muestra los bloques que ocupa archivo Þ "--block-size=TAMAÑO" nos permite establecer el tamaño de bloque que deseemos Þ "-h" muestra la información en unidades de bytes, KB, MB, GB
mkfs	Comando para crear un nuevo sistema de ficheros. Debido a la naturaleza del comando, se recomienda la lectura y la comprensión del manual del comando
dumpe2fs dispositivo	Muestra información del sistema de archivos indicado
fsck dispositivo	Hace un chequeo del sistema de archivos y repara sus posibles fallos. Es importante tener en cuenta que el dispositivo no debe estar montado en el sistema durante el uso del comando
sync	Sincroniza los archivos de *caché* y disco duro
badblocks dispositivo	Busca errores en la partición especificada Þ "-s" se "displaya" información del proceso

A.9. Usuarios y grupos

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
whoami	Muestra qué usuarios somos
groups	Muestra el grupo al que pertenecemos
who	Muestra los usuarios que están en el sistema en el momento de la ejecución del comando Þ "-T" indica, para cada usuario, si permite la recepción de mensajes
w [nombreUsuario]	Muestra información extensa sobre los usuarios del sistema en el momento de la ejecución. Si no se especifica un usuario en concreto, nos la muestra para todos
write nombreUsuario [consola]	Para mandar un mensaje al usuario especificado. Para determinar la consola, podemos utilizar los comandos "who" o "w"
talk nombreUsuario [consola]	Abre una sesión de *chat* con el usuario especificado si éste la acepta
mesg [y-n]	Sin pasar parámetros, muestra si está o no activa la recepción de mensajes. El parámetro "y" la activa y el parámetro "n" la desactiva
adduser [nombreUsurio]	Crea un nuevo usuario del sistema
userdel nombreUsuario	Elimina el usuario especificado del sistema
addgroup [nombreGrupo]	Crea un nuevo grupo
groupdel nombreGrupo	Elimina un grupo del sistema

A.10. Gestión de procesos

COMANDO	SIGNIFICADO Y PARÁMETROS ÚTILES
ps	Muestra los procesos del sistema Þ "-A" muestra todos los procesos que están ejecutándose en el sistema (igual que "-e") Þ "-H" muestra los procesos ordenados por jerarquía Þ "-l" muestra información extensa sobre los procesos
top	Muestra los procesos en ejecución de forma interactiva
kill PID	Manda una señal de finalización del proceso Þ "-9" envía la señal de matar el proceso
killall nombreProceso	Manda la señal de fin de proceso a todos los procesos indicados Þ "-9" envía la señal de matar los procesos
nice [comando]	Muestra el nivel de prioridad de ejecución que se utiliza por defecto, y si se le pasa como parámetro un comando junto con la prioridad deseada, ajusta la prioridad del comando a la especificada Þ "-n nuevoNivel" ejecutará el comando con la prioridad especificada
renice prioritad PID	Cambia la prioridad del PID especificado
at tiempo	Ejecuta el comando a la hora especificada Þ "-f comando" especifica el comando a ejecutar
atq	Muestra la cola de espera de "at"
atrm numComando	Elimina un comando de la cola de espera. El número de comando es el que devuelve "atq"
batch	Ejecuta los comandos o procesos cuando la carga del sistema es baja. Þ "-f comando" para especificar el comando

B. El editor vi

B.1. Introducción

Saber utilizar un editor de textos es imprescindible para poder editar y modificar los ficheros del sistema. Aunque existen centenares de editores diferentes, el **vi** siempre ha sido el editor por defecto de los sistemas like UNIX. Aunque en un principio el vi pueda parecernos un editor muy simple, a medida que nos vayamos acostumbrando a sus comandos veremos que tiene muchísimas utilidades que nos facilitan mucho la manipulación de los ficheros. Aunque para tareas largas (como cuando programamos) existen otros editores más útiles, la gran mayoría de administradores de sistemas utilizan el vi para muchas de las tareas de administración. El hecho de que se trate de un editor en modo texto (que permite su utilización en la consola del sistema) y estar disponible en todos los sistemas hacen del vi el editor ideal en los entornos UNIX.

Contenido complementario

En GNU/Linux se suele utilizar más el **vim** (Vi IMproved), que es 99,9% compatible con el vi pero añade unas cuantas funcionalidades más.

Para llamar el vi podemos utilizar alguno de los métodos que vemos en la siguiente tabla:

"`vi archivo`"	Edita el fichero en modo "`full screen`"
"`vi -r archivo`"	Recupera la última versión guardada del fichero (por los casos en que no salimos correctamente del editor y queda un fichero de *swap*)
"`vi + archivo`"	Edita el fichero y se sitúa en la última línea
"`vi +numLínea archivo`"	Edita el fichero y se sitúa en la línea indicada
"`vi archivo1 ...archivoN`"	Va editando todos los archivos especificados. Para saltar de uno a otro debemos escribir en modo comando "`:n`". Con "`:n!`" no guardamos las modificaciones
"`vi +/string archivo`"	Edita el fichero y sitúa el cursor en la primera ocurrencia del "`string`"

B.2. Modos del vi

El vi tiene dos modos de utilización: modo comando y modo inserción. En el modo comando todo lo que escribamos será interpretado

ANOTACIONES

por el editor para realizar acciones concretas, mientras que el modo inserción se utiliza para modificar el contenido del archivo. Cuando entramos en el vi, por defecto estamos en modo comando. Para cambiar a modo inserción, podemos utilizar cualquiera de las teclas de la siguiente tabla:

"a"	Inserta después del carácter donde estamos situados
"i"	Inserta antes del carácter donde estamos situados
"A"	Añade al final de la línea actual
"I"	Añade al principio de la línea actual
"R"	Entra en modo inserción reemplazando caracteres
"o"	Añade una línea en blanco debajo de la nuestra y pasa a modo inserción
"O"	Añade una línea en blanco encima de la actual y pasa a modo inserción

Para volver a modo comando, podemos utilizar la tecla ESC. En modo inserción lo único que podemos hacer es escribir texto, eliminarlo o desplazarnos con las teclas de AVPAG y REPAG. El modo comando nos permite muchísimas más acciones. En las siguientes tablas especificamos algunas de las más comunes:

Moviéndonos por el fichero	
"j"	Siguiente línea
"k"	Línea anterior
"l"	Siguiente carácter
"h"	Carácter anterior
"[["	Incio del archivo
"]]"	Final del archivo
"nG"	Ir a la línea "n"
"G"	Ir al final del archivo
RETURN	Siguiente línea
CTRL+F	Pantalla siguiente
CTRL+B	Pantalla anterior
CTRL+D	Media pantalla siguiente
CTRL+U	Media pantalla anterior

ANOTACIONES

Operaciones con archivos	
":w"	Guarda el fichero
":w nombreArchivo"	Guarda el fichero con el nombre indicado
":wq"	Guarda el fichero y sale del editor
":x"	Guarda el fichero y sale del editor
"ZZ"	Guarda el fichero y sale del editor
":q"	Sale si no ha habido cambios en el fichero
":q!"	Sale sin guardar los cambios
":e archivo"	Edita el archivo indicado si no hay cambios en el actual
":e! archivo"	Edita el archivo indicado perdiendo los cambios en el actual, si hubiera
":r archivo"	Añade el archivo indicado después de la línea actual
":Nr archivo"	Añade el archivo indicado después de la línea "N"
":sh"	Ejecuta un *shell* sin salir del editor; para salir del *shell* debemos escribir "exit"
":N,Mw!"	Guarda desde la línea "N" a la "M" descartando todas las otras
":N,M>>archivo"	Añade desde la línea "N" a la "M" el archivo indicado
":="	Muestra la línea actual
CTRL+G	Muestra el estado del fichero

Copiar, borrar, pegar, buscar y reemplazar	
"yy"	Copia la línea actual
"Nyy"	Copia las "N" líneas a partir del cursor
"p"	Pega las líneas copiadas debajo de la actual
"P"	Pega las líneas copiadas encima de la actual
"x"	Borra el carácter de debajo del cursor
"dw"	Borra la palabra de debajo del cursor
"dd"	Borra la línea actual
"D"	Borra desde la posición del cursor hasta el final de línea
"/string"	Busca el "string" a partir de la posición actual. Para continuar con la búsqueda, se puede utilizar "n" y "N" para ir hacia delante o atrás respectivamente
"?string"	Como el comando anterior pero en modo invertido
":set ic"	Realiza las búsquedas ignorando mayúsculas/minúsculas
":set noic"	Realiza las búsquedas con mayúsculas/minúsculas
":g/HOLA/s//ADIOS"	Sustituye todos los "HOLA" por "ADIÓS"

ANOTACIONES

En la línea inferior del editor veremos los comandos que vayamos escribiendo, que se ejecutarán al apretar el RETURN. Además, la mayoría de estos comandos permiten la repetición: tan sólo debemos escribir el número de veces que queremos que se ejecuten antes que el comando. Por ejemplo, con “`dd`” conseguimos que se borre la línea actual; si escribiéramos “`3dd`” en lugar de la línea actual, se borrarían las tres siguientes.

ANOTACIONES

C. Proceso de instalación de Red Hat Linux 9.0

C.1. Introducción

En este apéndice se pretende dar una idea general de los pasos básicos que hay que seguir para la instalación de Red Hat Linux 9.0. Se dan por sentados los conocimientos básicos adquiridos a lo largo del módulo y que han conducido a la instalación de Debian 3.0r1. Los conocimientos teóricos para abordar cualquiera de ambas instalaciones son prácticamente los mismos, por este motivo, este apéndice está completamente orientado a praxis, y en él sólo se resaltan las diferencias entre los dos procesos de instalación.

C.2. Inicio de la instalación

C.3. RHinicioinst

Para inicializar el proceso de instalación, basta con arrancar el ordenador desde CD-ROM y con el primer CD-ROM de la distribución dentro del dispositivo. Una vez terminado el proceso de arranque, se nos muestra una pantalla inicial con el logotipo de Red Hat Linux junto con una serie de opciones asociadas a las teclas desde F1 hasta F5, y se nos indica que si deseamos usar el modo gráfico para realizar la instalación, basta con pulsar INTRO para inicializarla o que, si deseamos hacerlo en modo texto, debemos escribir `text` y pulsar INTRO. Salvo casos excepcionales, siempre se escoge el modo gráfico, ya que hace más amena la instalación.

Antes de entrar en el modo gráfico, se nos preguntará si deseamos hacer un testeo del soporte físico de donde se extraerán los datos para la instalación. Si estamos seguros de que nuestros CD no están dañados en forma alguna, podemos omitir este proceso.

ANOTACIONES

A continuación se arranca anaconda, el programa base de la instalación. Transcurridos unos segundos, entramos en la pantalla de bienvenida del programa. A partir de este momento, la pantalla estará dividida en dos áreas. En la de la izquierda se nos informa del punto de la instalación donde nos hallamos y se nos suministra información relativa al mismo; a la derecha se nos muestran las opciones disponibles. En todo momento dispondremos del ratón para hacer las selecciones que estimemos oportunas.

C.4. Primeros aspectos

En primer lugar, debemos seleccionar el idioma que se usará durante el proceso de instalación. Esta selección no afectará en ningún aspecto a la instalación resultante. En segundo lugar, hay que configurar el teclado. Este aspecto sí que es importante de cara al resultado de la instalación. El tercer punto es la configuración del ratón.

C.5. Tipo de instalación

Llegados a este punto, se nos pregunta por el tipo de sistema de instalación que queremos hacer. Probablemente, en la mayoría de los casos la mejor opción es la última que se nos ofrece, es decir, la personalizada, ya que, tal como se nos indica, esto nos permitirá tener el máximo control del proceso de instalación. El resto de opciones son simplificaciones de esta opción y, en consecuencia, se traducen directamente en una pérdida de control acerca de lo que se está haciendo.

C.6. Partición del disco duro

El sistema de instalación nos propone por defecto la partición del disco duro. Alternativamente, se nos ofrece la posibilidad de utilizar Disk Druid para realizar nosotros mismos esta tarea. Es recomendable optar por esta última opción, ya que el programa es fácil de utilizar, y además nos permitirá estructurar la partición a la medida de nuestras necesidades. Si ya disponemos de una partición *swap*,

aunque sea de otro sistema, no necesitaremos crear una nueva para esta instalación, puesto que podremos usar la existente, ya que en las particiones de tipo *swap* sólo se guarda información de tipo volátil.

C.7. Gestor de arranque

En este caso, Red Hat Linux nos propone como gestor de arranque Grub. Si ya disponemos de un gestor de arranque en nuestro equipo, no es necesario, a no ser que deseemos reemplazarlo, proceder a la instalación de Grub. Bastará con adecuar la configuración de arranque para que nos permita acceder al nuevo sistema operativo.

C.8. Configuración de dispositivos

El programa de instalación detectará la mayor parte del hardware que tengamos instalado, y llegados a este punto, se nos muestran las pantallas pertinentes para su configuración.

Por lo que a la configuración de red se refiere, el programa de instalación nos preguntará por la gestión de la seguridad, basándose en la configuración de un *firewall*. Si optamos por utilizarlo, deberemos especificar los puertos a los que se deja acceso externo. Es un tema que deberemos tener siempre presente, ya que el resto de puertos estarán cerrados. Así pues, si configuramos algún servicio que use un puerto que no hayamos especificado en este apartado, deberemos abrirlos manualmente *a posteriori*.

C.9. Configuración idiomática

En este apartado sí que es importante que especifiquemos cuál es el idioma del mapeado del teclado, ya que esto nos permitirá tener acceso a todos sus caracteres. A continuación, se nos mostrará un mapa mundi donde deberemos especificar nuestro posicionamiento geográfico mediante el ratón.

ANOTACIONES

C.10. Política de *passwords*

En primer lugar se nos pedirá que introduzcamos el *password* de *root*, el cual deberá contener como mínimo seis caracteres. Una vez hecho esto, se nos mostrará que tanto el sistema de *passwords* md5 como el de shadow se instalarán, a no ser que especifiquemos lo contrario, y se nos permitirá establecer las configuraciones de los sistemas NIS, LDAP, Kerberos 5, y Samba (para un sistema *stand alone*, ninguno de estos sistemas debe ser configurado).

C.11. Selección de aplicaciones

A continuación, debemos seleccionar las familias de aplicaciones que queremos instalar, y dentro de ellas, qué aplicaciones deseamos que se instalen en nuestro sistema. Una vez más se insta a que no se instalen masivamente aplicaciones, ya que esto va claramente en detrimento del rendimiento del sistema, sino que sólo se instalen aquellas que sabemos de forma certera que vamos a utilizar, y que vayamos instalando el resto de aplicaciones a medida que las necesitemos. Sí que es recomendable, si se van a usar, dejar las selecciones que están hechas para el sistema gráfico. Una vez terminada la selección, se iniciará la instalación de todo el sistema: formateo de particiones, configuración de dispositivos e instalación de paquetes, a partir de la información que se ha facilitado. Si no disponemos de suficiente espacio en el disco duro para que se instalen todas las aplicaciones, se nos devolverá al entorno de configuración, para que, o bien deseleccionemos algunas de las aplicaciones, o para que asignemos más espacio a la partición pertinente (normalmente, la que alberga el directorio `/home`, que es en la que se instalan la mayoría de datos –para refraccionar el disco, habrá que volver a inicializar el proceso de instalación).

C.12. *Boot disk*

Cuando el proceso de transferencia de datos ha terminado, automáticamente se arranca el proceso de postinstalación, se instala el *boot loader*, si es el caso, y una vez han finalizado estos procesos auto-

ANOTACIONES

máticos, se nos pregunta si deseamos crear un *boot disk*, el cual puede ser usado para arrancar la nueva instalación, si no hemos instalado el *boot loader* o no hemos configurado el que ya teníamos instalado correctamente.

C.13. Configuración del sistema gráfico

En la mayoría de casos, llegados al apartado de configuración del sistema gráfico (este apartado se puede omitir y realizar a la configuración manualmente mediante el sistema de nuestra elección), si es que hemos seleccionado sus paquetes para ser instalados, el sistema de instalación detectará automáticamente el tipo de tarjeta de vídeo y nos propondrá el *driver* a utilizar; lo mismo sucederá con el monitor. Finalmente, deberemos especificar la resolución con la que queremos trabajar, y la profundidad de color. Asimismo, se nos pedirá que especifiquemos qué tipo de *login* deseamos utilizar, si el gráfico, opción por defecto, o bien el de modo texto.

C.14. Últimos pasos

Aquí termina la instalación del sistema operativo y de todas las aplicaciones seleccionadas durante el proceso. Basta con "rebootar" el sistema para que la primera vez que lo arranquemos se nos pregunte si deseamos utilizar `kudsu` para proceder a la configuración del hardware presente.

En primer lugar se nos sugiere la creación de una cuenta de usuario estándar para evitar el uso de *root* indiscriminadamente. Tras la configuración de fecha y hora, se nos preguntará si deseamos registrar nuestro sistema operativo, y a continuación, si lo deseamos, podremos proceder a la instalación de más paquetes. Es el momento de instalar aquellos paquetes que están dentro de la distribución oficial, y que deseamos utilizar. Una vez finalizada la instalación de paquetes adicionales, tenemos el sistema totalmente operativo y listo para ser utilizado. Se recomienda la lectura de http://rpm.redhat.com/RPM-HOWTO/, ya que Red Hat Linux usa su propio sistema de paquetes.

ANOTACIONES

D. Herramientas de administración

D.1. Introducción

Cuando se administra un sistema GNU/Linux es necesario conocer una gran variedad de aplicaciones y programas diferentes. Aunque antes de instalar cualquier aplicación es totalmente imprescindible leerse detenidamente la documentación que incorpora, en algunos casos las configuraciones pueden llegar a ser realmente complejas. Por este motivo, desde hace ya varios años han ido apareciendo herramientas de administración más intuitivas que permiten manejar múltiples aplicaciones y servicios de forma más amena.

Generalmente, estas herramientas globales de administración incorporan mecanismos para poder configurar y manejar los aspectos básicos del sistema y los ficheros de configuración de las aplicaciones que utilizamos. Si bien es interesante saber que existen estas herramientas, no es recomendable que basemos toda la configuración de un servidor en ellas por varios motivos. En primer lugar, debemos tener en cuenta que estos programas no siempre tienen en cuenta todas las posibilidades que los servidores proporcionan. Esto puede provocar que dejemos sin una configuración adecuada alguna opción importante para nuestras necesidades, que no tengamos en cuenta algún sistema de seguridad, etc. En segundo lugar, tampoco podemos olvidar que aunque el entorno de configuración sea más ameno y, generalmente, más fácil de utilizar y manejar, debemos conocer qué es lo que realmente se hace cuando activamos las opciones de los diferentes programas y servicios que configuramos. Aunque el entorno sea muy intuitivo, esto no implica que no debamos saber qué significa exactamente cada opción. Si no tenemos un conocimiento extenso del servicio que estamos manipulando, es muy fácil generar errores que pueden provocar un mal funcionamiento del sistema, agujeros de seguridad, etc. Finalmente, otro motivo para no utilizar únicamente estas aplicaciones es que en algún momento el sistema puede tener fallos o errores que no nos permitan utili-

Contenido complementario

Cuando instalamos alguna herramienta general de administración, es muy importante que limitemos su uso y acceso a sólo el *root* del sistema, de lo contrario cualquier usuario podría modificar cualquier aspecto del sistema. Además, también debemos estar muy alerta de los agujeros de seguridad que pueden aparecer en ellas, ya que al tener que manejar los programas instalados en el sistema, la mayoría de estas herramientas deben ejecutarse con los permisos de *root*, con el peligro que esto supone.

ANOTACIONES

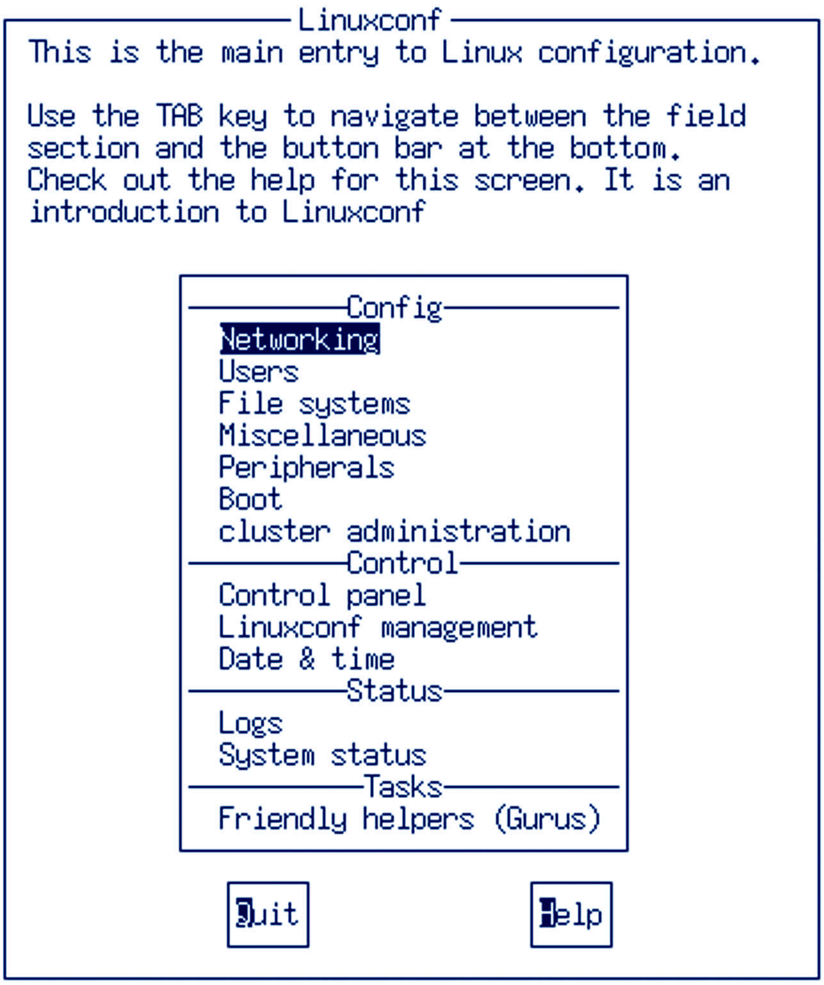
zarlas o sencillamente que en otros sistemas que tengamos que administrar no estén instaladas. Si no conocemos con un poco de detalle los ficheros de configuración de las aplicaciones que utilizamos, nos encontraremos totalmente indefensos ante cualquier pequeño problema que pueda surgir.

Por todo ello, la utilización de estas herramientas debe realizarse con cuidado y sabiendo exactamente qué es lo que están modificando. En algunos casos pueden ser muy útiles para ver cómo realizar algún tipo de configuración complicada o para detectar errores que hayamos generado. La forma como deberíamos utilizarlas debe servirnos a modo de complemento de nuestra administración, pero nunca basarnos totalmente en ellas.

Todas las distribuciones de GNU/Linux acostumbran a incorporar sus propias herramientas automáticas de administración. Esto es una característica claramente diferenciadora de las distintas distribuciones de GNU/Linux. En SuSE, por ejemplo se incorpora una aplicación denominada `Yast2`, que nos permite realizar casi cualquier operación de configuración del sistema; RedHat incorpora múltiples programas diferentes para configurar la red, los daemons, los servidores de aplicación, etc.; al instalar un paquete en Debian, ya se permite inicializar una configuración a partir de las respuestas que damos en varias pantallas de diálogo; algunas aplicaciones llevan sus propios *scripts* para permitir configuraciones estándar más rápidas, etc. Aun así, si sabemos qué es lo que realmente hacen estas aplicaciones y en qué ficheros guardan su configuración, al tener problemas con el sistema su arreglo será mucho más fácil. Además de estas herramientas únicas para la distribución que utilicemos, existen otras generales que podemos instalar en la mayoría de distribuciones existentes. Aunque hay unas cuantas decenas y cada administrador debe escoger la que más le guste o se adapte a sus necesidades, en este apéndice mostraremos un par de las más versátiles y populares: `linuxconf` y `webmin`.

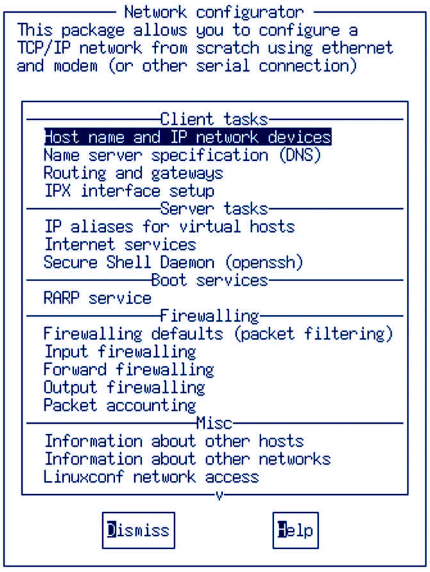
D.2. Linuxconf

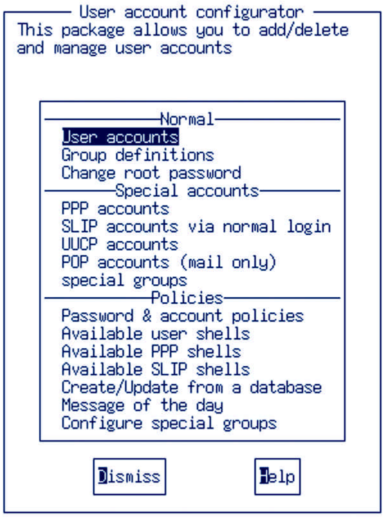
La aplicación de administración general `Linuxconf` está basada en un entorno de menús de texto que podemos utilizar desde cualquier consola del sistema. En la siguiente figura podemos ver el menú principal:



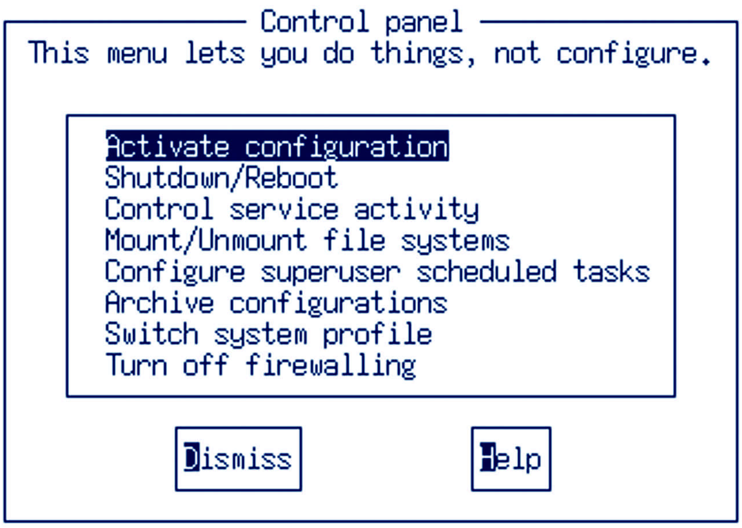
Como podemos apreciar en la imagen, `linuxconf` divide sus operaciones en las siguientes secciones:

- Configuración: ésta es la sección principal de `linuxconf`, donde podemos configurar la mayoría de aspectos del sistema como la red, los usuarios, los periféricos instalados, etc. En las siguientes figuras podemos ver el diálogo de configuración de la red y de usuarios:





- Control: sección para realizar acciones concretas en el sistema tales como montar o desmontar unidades, cambiar la hora del sistema, personalizar los menús, etc. En la siguiente figura podemos apreciar algunas de las acciones del menú de panel de control:



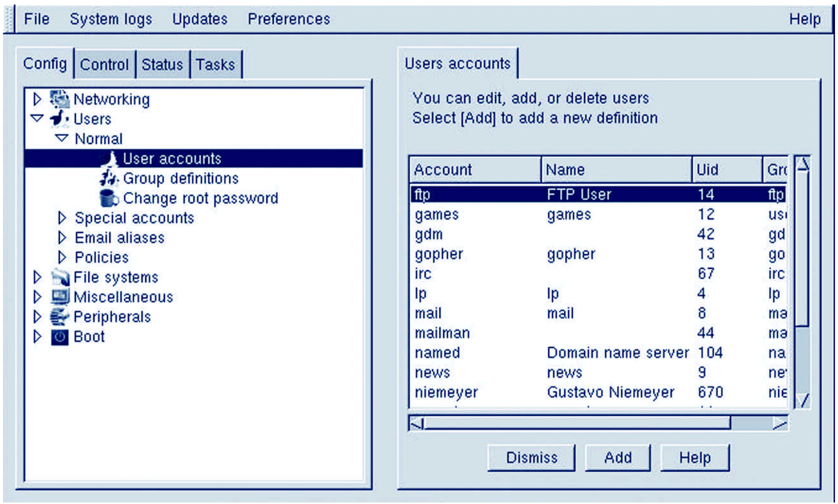
- Estado: cuando queramos ver los logs o el estado de algún aspecto del sistema, podemos recurrir a los menús de esta sección. En ellos se utilizan muchos de los comandos básicos del sistema para ver el estado del disco, la memoria usada, etc.

- Tareas: otros diálogos de configuración para inicializar correctamente un módem, la red, etc.

ANOTACIONES

Otra forma de utilizar este programa es a partir de un navegador web. Por defecto, el acceso vía navegador está cerrado, por lo que antes de utilizarlo deberemos habilitarlo a partir del menú de `Networking`, `Linuxconf network access` y activando la opción de `Enable network access`. Abriendo el navegador y accediendo a http://localhost:98/ tendremos los mismos diálogos y opciones del menú de `linuxconf` en formato html. Por defecto, sólo se podrá acceder a este servicio desde la misma máquina, aunque es recomendable activarlo solamente cuando lo queramos utilizar.

Finalmente, otro proyecto relacionado con `linuxconf` es el `gnome-linuxconf`, que tiene las mismas funciones que veíamos anteriormente pero se puede utilizar en las X. En la siguiente figura podemos ver su aspecto:



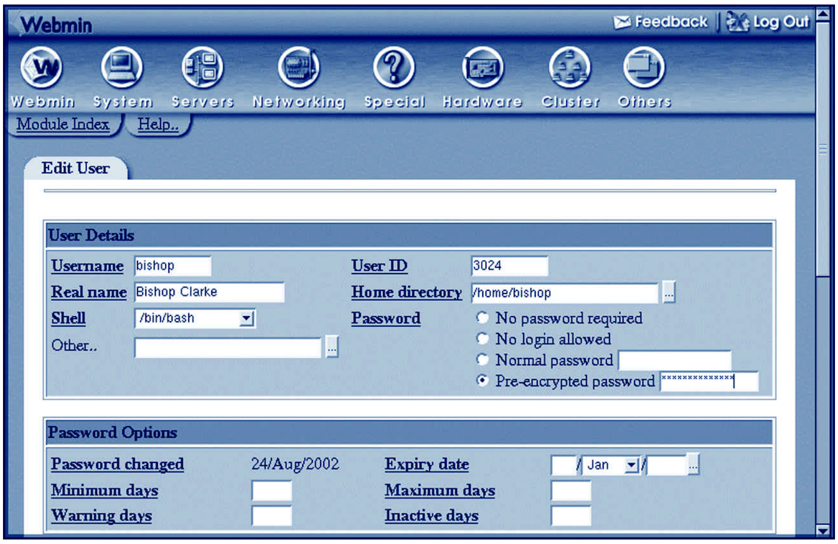
D.3. Webmin

Tal como su nombre indica, `webmin` es una herramienta de configuración vía web. Su diseño está muy bien conseguido y la mayoría de sus métodos de configuración están muy bien pensados. Bien utilizada, es una herramienta que puede llegar a sernos muy útil. Por defecto, al instalarla abre el puerto 10000 para que podamos acceder a ella a partir de un navegador cualquiera. Antes de entrar nos pedirá la contraseña del administrador del sistema, aunque también tiene un sistema muy útil de administración de usuarios propios, por

medio del cual podemos especificar qué acciones puede realizar cada uno de ellos. Esta opción es muy interesante porque permitirá configurar más de un administrador de sistema cada uno especializado en algunas tareas.

Para hacernos una idea de la aplicación, a continuación mostramos una serie de capturas de diferentes secciones:

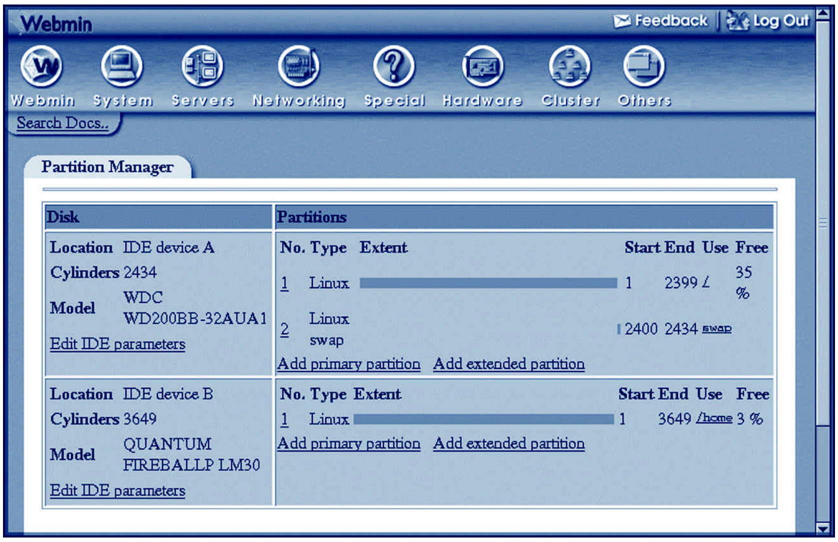
- Administración de usuarios del sistema:



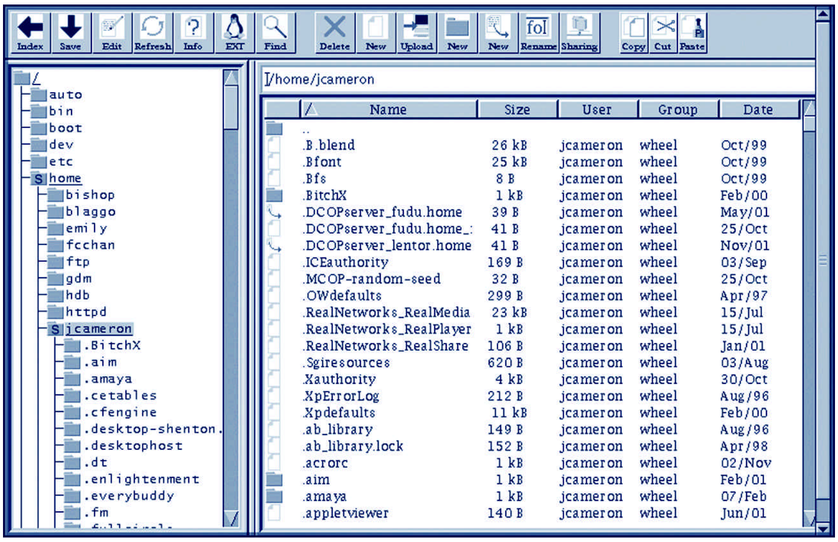
- Configuración de arranque del sistema:



- Información de disco:



- Navegador de archivos:



ANOTACIONES

UOC

La universidad virtual

Formación de Posgrado